UNIVERSIDADE FEDERAL DO PARANÁ
DEPARTMENTO DE ENGENHARIA ELÉTRICA
CURSO DE ENGENHARIA ELÉTRICA

Érica de Oliveira Gomes
Helder Henrique Avelar

ALARME AUTOMOTIVO INTELIGENTE

Curitiba
2012

Érica de Oliveira Gomes
Helder Henrique Avelar

ALARME AUTOMOTIVO INTELIGENTE

Trabalho de Conclusão de Curso de Engenharia Elétrica, Departamento de Engenharia Elétrica, Setor de Tecnologia, Universidade Federal do Paraná.

Orientador: Prof. PH.D. André Augusto Mariano

Curitiba
2012

Érica de Oliveira Gomes
Helder Henrique Avelar

ALARME AUTOMOTIVO INTELIGENTE

TRABALHO APRESENTADO AO CURSO DE ENGENHARIA ELÉTRICA DA UNIVERSIDADE FEDERAL DO PARANÁ COMO REQUISITO À OBTENÇÃO DO TÍTULO DE GRADUAÇÃO.

COMISSÃO EXAMINADORA

_____

PROF. PH.D. ANDRÉ AUGUSTO MARIANO

_____

PROF. DR. EDUARDO PARENTE RIBEIRO

_____

PROF. ROBERTO HEINRICH

CURITIBA, JANEIRO DE 2012.

AGRADECIMENTOS

Nossos sinceros agradecimentos a todos que nos inspiraram, guiaram e ofereceram ajuda durante este projeto e por todos os anos de Universidade. Agradecimentos especiais às nossas famílias, que por todos os últimos anos foram pilares de sustentação e sem as quais nada seria possível. Um singelo “obrigado” aos bons e duradouros amigos que tornaram muitas coisas mais divertidas e prazerosas, inclusive a realização do trabalho aqui presente. Agradecimentos àqueles que amamos e nos inspiram e motivam em particular. Por fim, sinceros agradecimentos a Deus por Sua presença e conforto.

RESUMO

No ano de 2010, mais de 77 (setenta e sete) milhões de veículos motores foram produzidos em todo o mundo[1]. Diretamente proporcional ao mercado automobilístico em si, crescem os fornecedores de facilidades, comodidades, segurança e demais 'acessorizações' automobilísticas, assim como os alarmes. Neste trabalho de conclusão de curso está apresentada uma solução moderna e inteligente de alarme automotivo, capaz de comunicar com o proprietário, através da rede de celulares, em caso de violação do alarme e movimentação indevida do veiculo no qual estiver instalada. Para o desenvolvimento do projeto foram utilizados um alarme de carro comum, módulo GPS ME1000-RW e módulo GSM SIM340CZ. O controle dos módulos e da interface entre eles foi feito utilizando micro controlador MPC89E515A, da família 8051, programado em *Assembly*. A comunicação entre o *hardware* e servidor foi realizada através de sinal GPRS, com protocolo TCP/IP. Do lado do *software*, foram utilizados um servidor *Web Apache* e banco de dados *MySQL* do departamento de Engenharia Elétrica da UFPR, programados em PHP. Por fim, a interface com o cliente se dá por um web site, programado em HTML, CSS, PHP e *JavaScript*. Estão apresentadas neste relatório as etapas de desenvolvimento e resultados dos testes, bem como introduções às tecnologias utilizadas e os códigos relevantes programados ao longo do trabalho.

ABSTRACT

In the year of 2010, roughly 77 (seventy seven) million motor vehicles were manufactured all over the globe[1]. In a direct proportion to the automobilist market itself follows the growth of facilities, commodities, security and overall automobile “acessorizing” providers, therein included the section of car alarms. Presented in this final project is a modern and intelligent car alarm solution, capable of communicating with the proprietary through cell phone network in case of violation of the alarm and unauthorized move of the vehicle in which it is installed. In order to develop the project, a simple car alarm, GPS ME1000-RW and GSM SIM340CZ modules were used. The control and interface between modules was done using an 8051 family MPC89E515A processor, programmed in Assembly. The communication between hardware and server was done through GPRS signal, with TCP/IP protocol. On the software side, the project made use of an Apache Web server and a MySQL database provided by UFPR Electrical Engineering department, both programmed in PHP. Lastly, the client interface is done by a web site, programmed in HTML, CSS, PHP and JavaScript. Presented in this report are the development stages and test results, as well as brief introductions to the used technologies and the relevant codes programmed throughout the project.

LISTA DE FIGURAS

Figura 1 - Pesquisa preliminar sobre alarmes automotivos simples. 14
Figura 2 – Topologia do sistema 16
Figura 3 - Módulo GPS 1000RW 22
Figura 4 – Gravador TH96A para uCs Megawin. 24
Figura 5 - Macro-estrutura de uma página HTML. 29
Figura 6 – Exemplo de formulário HTML. 31
Figura 7 - Circuito de alimentação do módulo GSM. 34
Figura 8 - Circuito da placa do módulo GSM. 35
Figura 9 - Fotos da placa de controle do módulo GSM vista de cima e de baixo. 36
Figura 10 – Fotos do kit de desenvolvimento para SIM300 36
Figura 11 - Modulo GSM com antena ligada ao pad. 37
Figura 12 - I/O Ninja com sniffer de uma conexão realizada com sucesso. 38
Figura 13 - Fotos da placa para gerenciamento 40
Figura 14 - Esquemático da placa de controle do sistema 41
Figura 15 - Fluxograma representando o funcionamento do firmware do projeto. 42
Figura 16 – Tabelas criadas no banco de dados MySQL. 43
Figura 17 – Fluxograma do portal Web. 45
Figura 18 – Pagina 14 do manual do modulo GPS ME1000-RW (ANEXO 2) 47
Figura 19 – Pagina inicial do portal AAI. 55
Figura 20 – Página de cadastro de novo usuário. 55
Figura 21 – Página de acesso restrito ao usuário autenticado. 55
Figura 22 – Listagem de carros cadastrados de um usuário. 56
Figura 23 – Página de rastreamento de posições com pontos no mapa. 56
Figura 24 – Página de rastreamento de posição com listagem simples. 57

LISTA DE TABELAS

Tabela 1 – Custo total do módulo do alarme AAI 52
Tabela 2 – Comparação de funcionalidades do módulo AAI com os concorrentes 53

LISTA DE ABREVIATURAS E SIGLAS

AAI – Alarme Automotivo Inteligente

ANATEL - Agência Nacional de Telecomunicações

API – *Application Programming Interface*

APN – *Access Point Name*

bps – bits por segundo

Cartão SIM – *Subscriber Identification Module*

CI – Circuito Integrado

CPU – *Central Processing Unit*

CSS – *Cascading Style Sheets*

DB – *Data Base* (ou “Banco de dados”)

DNS – *Domain Name System*

FTP – *File Transfer Protocol*

GPGGA – *Global Positioning System Fix Data*

GPGSA – *Global Positioning System Fix Data*

GPGSV – *GPS Satellites in view*

GPS – *Global Positioning System*

GPRMC – *Recommended minimum specific GPS/Transit data*

GPRS – *General packet radio service*

GPVTG – *Track made good and ground speed*

GSM – *Global System for Mobile communication*

HTML – *HyperText Markup Language*

HTTP – *Hypertext Transfer Protocol*

IMEI – *International Mobile Equipment Identity*

JAVA – Linguagem de programação orientada a objeto

LVTTL ou TTL – *Low Voltage Transistor-Transistor Logic*

MySQL – Sistema de gerenciamento de bancos de dados

NMEA – *National Marine Electronics Association*

OSI – *Open Systems Interconnection*

PDP – *Packet Data Protocol*

PHP – *Hypertext Preprocessor*

RAM – *Random Access Memory*

SMS – *Short Message Service*

SPS - *Standard Positioning Service*

SQL – *Structured English Query Language*

TCP/IP – *Transmission Control Protocol/Internet Protocol*

UART – *Universal Asynchronous Receiver/Transmitter*

uC – Micro controlador

UFPR – Universidade Federal do Paraná

SUMÁRIO

AGRADECIMENTOS....10
RESUMO....10
LISTA DE ABREVIATURAS E SIGLAS....2
SUMÁRIO....9
1. INTRODUÇÃO....12
1.1. PROBLEMA E MOTIVAÇÃO....13
1.2. OBJETIVOS....15
1.2.1. OBJETIVO GERAL....15
1.2.2. OBJETIVOS ESPECIFICOS....16
1.2.2.1. DISPOSITIVO ELETRÔNICO....16
1.2.2.2. SERVIDOR....17
1.2.2.3. INTERFACE COM O CLIENTE....17
2. FUNDAMENTAÇÃO TEORICA....19
2.1. GSM E GPRS....19
2.1.1. GSM....19
2.1.2. GPRS (*General packet radio service)*....19
2.1.3. ESCOLHA DO MÓDULO....20
2.2. GPS....21
2.2.1. FUNCIONAMENTO....21
2.2.2. MÓDULO ESCOLHIDO....21
2.2.3. O PADRÃO NMEA 0183....22
2.3. MICROCONTROLADOR....22
2.3.1. FIRMWARE....23
2.4. COMUNICAÇÃO SERIAL....24
2.5. I/O NINJA....25
2.6. PROTOCOLO TCP/IP....25
2.7. SERVIDOR....25
2.7.1. SERVIDOR *WEB APACHE*....26
2.7.2. SERVIDOR BANCO DE DADOS *MYSQL*....26
2.7.3. EASY PHP....26

2.8. PROTOCOLO HTTP 27
2.9. PROTOCOLO FTP 28
2.10. HTML E CSS 28
2.10.1. HTML Pad 29
2.11. PHP 30
2.12. API GOOGLE MAPS 32
3. MÉTODOS 33
3.1. DISPOSITIVO ELETRÔNICO 33
3.1.1. MÓDULO GSM 33
3.1.2. MÓDULO GPS 39
3.1.3. PLACA PARA GERENCIAMENTO 39
3.1.4. FIRMWARE 40
3.2. SERVIDOR 43
4. MATERIAIS 48
4.1. CIRCUITO 48
4.2. SERVIDOR 48
5. ANALISE DOS RESULTADOS 50
5.1. HARDWARE 50
5.1.1. ANÁLISE DE CUSTO 51
5.2. WEBSITE 54
6. CONCLUSÃO 59
6.1. MELHORIAS FUTURAS 59
7. REFERENCIAS 61
8. ANEXOS 66
8.1. ANEXO A – Lista de comandos AT utilizados no projeto. 66
8.2. ANEXO B – Guia do usuário KIT SIM900 66
8.3. ANEXO C – *Datasheet* Módulo GPS 1000ME-RW 67
8.4. ANEXO D – *Datasheet* microprocessador MPC89E515A 67
8.5. ANEXO E – Grupo de comandos AT para SIM300 67
8.6. ANEXO F – *Hardware Design* SIM 300 67
8.7. ANEXO G – *Código do Website* 67
INDEX.PHP 67

INDEX.CSS 69
MAIN.PHP 73
CADASTRO_USUARIO.PHP 75
RECEBE_CADASTRO.PHP 77
RECEBE_LOGIN.PHP 80
LOGADO_INDEX.PHP 82
CADASTRO_VEICULO.PHP 84
LISTAR_VEICULO.PHP 86
RASTREAR_VEICULO.PHP 88
LISTAR_POSIÇÃO.PHP 92
LOGOFF.PHP 96
8.8. ANEXO H – *Código Gravado no Microcontrolador* 96

1. INTRODUÇÃO

Alarmes automotivos são dispositivos utilizados em veículos com o intuito de desencorajar ou evitar que objetos no interior do carro, ou o próprio carro, sejam roubados. Os alarmes utilizam uma combinação de técnicas, sendo as mais comuns os sensores de movimento e os detectores de luz de teto. Uma vez que o alarme sonoro seja acionado, este só poderá ser cessado pelo controle remoto.

Muitos dos atuais sistemas de segurança fornecem proteção em várias zonas, por exemplo, podendo ser colocados sensores mecânicos no capô ou porta-malas. A versatilidade do mercado de alarmes permite que o cliente decida o que o alarme deve proteger em seu veículo. Muitos sistemas de alarme integram-se com demais funcionalidades do carro - o sensor de ligar e desligar do controle remoto muitas vezes é multifuncional, ligando e desligando também as travas e vidros elétricos.

Embora sejam vastamente vendidos, a eficácia dos alarmes sonoros é freqüentemente contestada [2] devido à banalização do mesmo e a freqüência com a qual este é falsamente disparado. Estudos Norte Americanos afirmam que a maioria da população civil não dá atenção a um alarme automotivo, por suspeitarem tratar-se de um disparo não intencional por parte do dono do veículo ou por não terem interesse em se envolverem com um caso de crime urbano. No entanto, o entendimento da palavra “eficácia” não deveria estar atrelado ao impedimento do roubo, uma vez que o escopo de um alarme automotivo simples é somente o de afastar o bandido, impondo uma barreira a mais do que a de um carro desprotegido, e não o de impedir completamente o furto.

Com o intuito de impedir o furto, outros dispositivos vieram ao mercado. Dentre eles a trava de volante, capaz de impedir o giro de volante a não ser que o dono do veículo a destrave, ou imobilizadores, capazes de evitar o ligamento do motor sem que o dispositivo, geralmente escondido em uma área do carro escolhida pelo dono, seja pressionado.

Além desta categoria de produtos, alarmes com rastreamento também passaram a ser vendidos. Estes, assim como o alarme simples e sonoro, não são capazes de impedir o furto, mas mostram-se vantajosos por poderem proporcionar a localização do veículo e facilitar a captura e devolução do mesmo para o dono. Alarmes com rastreamento não costumam ser dispositivos de fábrica ou oferecidos pelas montadoras, sendo, em sua maioria, oferecidos por empresas privadas de segurança, com custos mensais agregados de monitoramento que encarecem bastante o produto. [3]

Tendo em vista as opções mais populares de alarmes no mercado atual, este trabalho foi desenvolvido com o intuito de apresentar uma solução diferenciada. O produto aqui apresentado se trata de um alarme sonoro com rastreamento, diferencial por ser capaz de comunicar a violação ao dono do veiculo em tempo real, além de ser independente de qualquer empresa de monitoramento. O Alarme Automotivo Inteligente (AAI) desenvolvido neste projeto apresenta baixo custo, é capaz de comunicar rapidamente a violação dos sensores e permite com que o cliente rastreie seu veículo autonomamente em caso de movimentação indevida, podendo acionar a polícia ou agir como julgar necessário.

1.1. PROBLEMA E MOTIVAÇÃO

Ao longo da concepção desde projeto, o principal pré-requisito era a utilidade comercial do mesmo – uma vez ciente da vasta quantidade de veículos produzidos anualmente no país e da forte fatia do mercado ocupada pelos diversos modelos de alarmes automotivos, tornou-se fácil visualizar o potencial comercial do produto a ser desenvolvido.

Partindo de uma pesquisa de opinião simples realizada nas primeiras etapas do projeto, foi possível perceber que muitos dos proprietários de carros que possuem alarmes automotivos simples, não se satisfazem com a segurança que este os proporciona pois ainda que seus carros sejam violados e os alarmes acionados, dificilmente eles estarão próximos o suficiente para verificarem a ocorrência.

Pode-se verificar na figura 1 o resultado da pesquisa feita com aproximadamente 30 (trinta) alunos da Universidade Federal do Paraná, onde eram abordadas as seguintes perguntas:

- Seu carro possui alarme? (Sim/Não)
- Seu alarme é modelo simples (sonoro)? (Sim/Não)
- Você acredita que seu alarme é eficaz/suficiente? (Sim/Não)

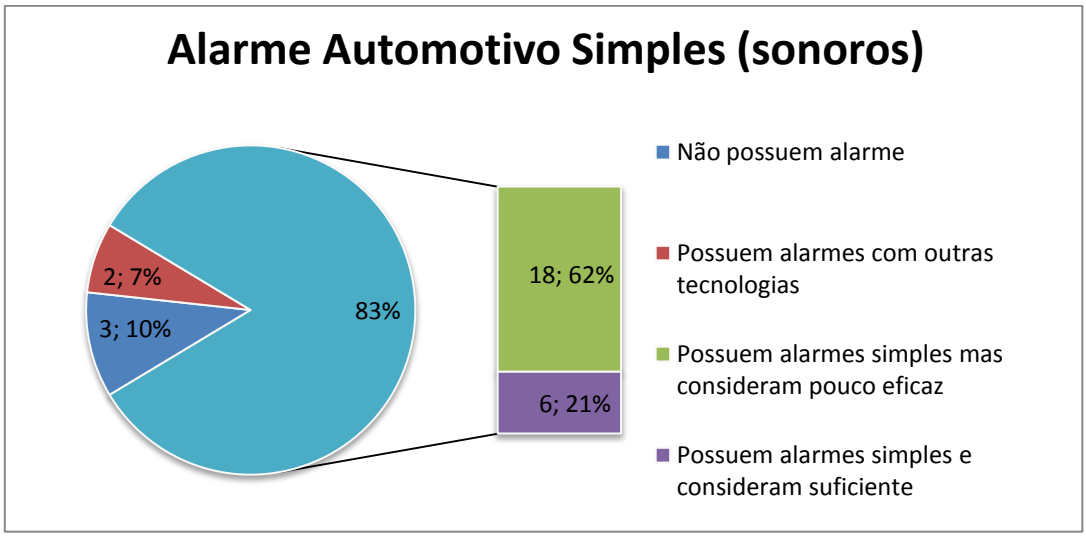


Figura 1 - Pesquisa preliminar sobre alarmes automotivos simples.

Dos alunos entrevistados que possuíam carro, 2,7% não tinham qualquer alarme instalado, 3,10% possuíam alarmes com outras tecnologias e a maioria possuía alarme sonoro simples. Esta ultima parcela foi dividida ainda em outras duas, os que acreditavam na eficácia de seus alarmes e os que o achavam insuficiente.

Observando que 18 pessoas, representando 62% dos entrevistados, estavam insatisfeitas com seus sistemas de alarme, cresceu a motivação em realizar um projeto neste contexto. Para solucionar a falta de comunicação entre o alarme simples e o proprietário e proporcionar uma sensação um pouco maior de segurança. O sistema proposto neste trabalho possui a capacidade de comunicar-se em tempo real com o celular do proprietário no caso de violação do alarme ou movimentação indevida do veículo, além de permitir o rastreamento do automóvel.

Visando baixar custos e proporcionar autonomia do cliente ao lidar com o sistema, a empresa fictícia fornecedora dos Alarmes Automotivos Inteligentes irá prover um *website* onde cada cliente poderá, de forma autônoma, cadastrar os carros instalados com os alarmes em questão e rastreá-los em caso de movimentação indevida. Desta forma, não só serão reduzidos os custos como será aumentada a confiança do cliente na utilização das informações de posição do seu veículo, pois somente o mesmo, com sua senha, terá acesso a estas informações.

De forma a produzir o sistema proposto, tecnologias amplamente utilizadas como GPS e GSM, além de sistemas e linguagens de programação fortemente presentes no mercado, como Assembly, MySQL, PHP e Java, foram opções escolhidas. Parte da motivação para a escolha deste tema se tratava da gama de tecnologias que poderiam ser aprendidas ou dominadas, com o intuito de consolidar os conhecimentos adquiridos ao longo da graduação e ampliar outros conceitos pertinentes.

1.2. OBJETIVOS

1.2.1. OBJETIVO GERAL

O objetivo geral deste trabalho é o desenvolvimento de um alarme automotivo inteligente, contemplando *hardware* e *software*, capaz de comunicar violação dos sensores e movimentação indevida do veículo com o proprietário, através de mensagens *SMS* para celular e envio de pacotes *GPS* para um servidor dedicado.

A Figura 2 apresenta um esquemático do trabalho como um todo, em suas três etapas principais: O alarme a ser instalado no veículo, o servidor e as interfaces de contato com o cliente.

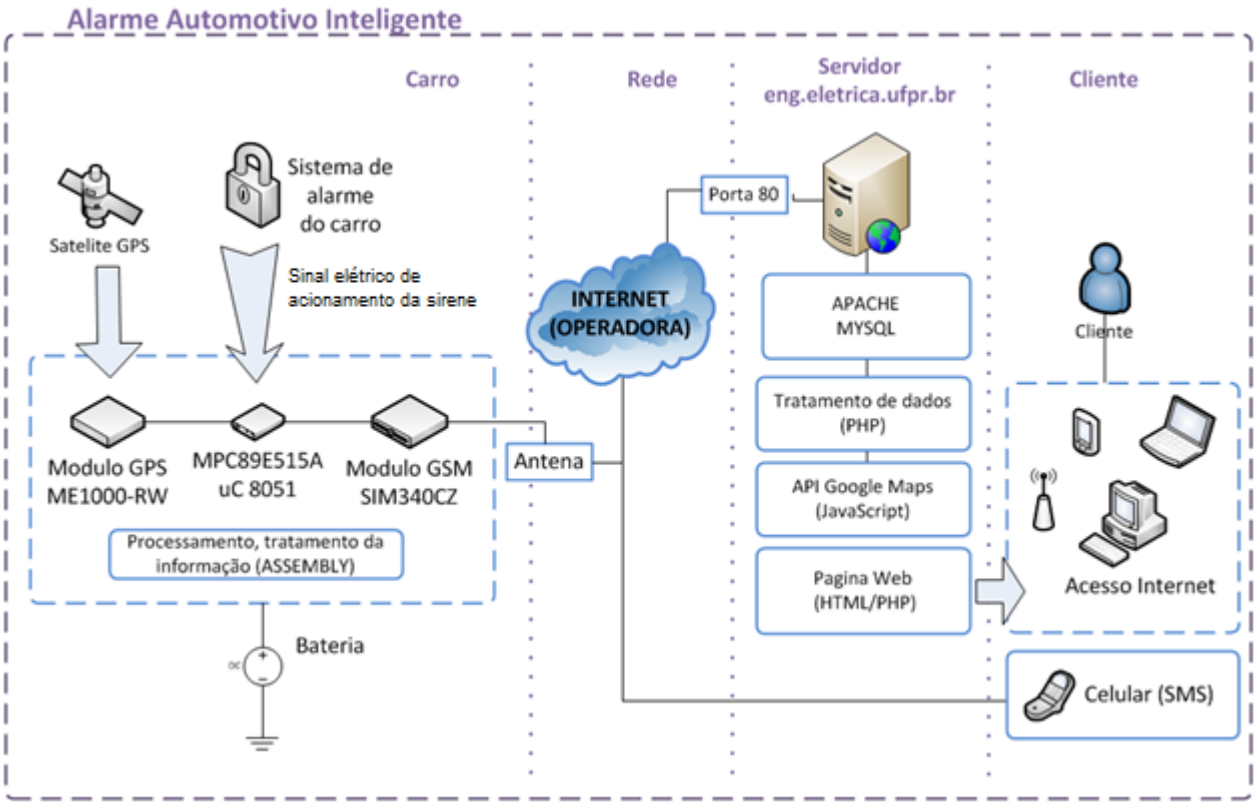


Figura 2 – Topologia do sistema

1.2.2. OBJETIVOS ESPECIFICOS

1.2.2.1. DISPOSITIVO ELETRÔNICO

a) Desenvolvimento de uma placa de alimentação para o módulo GSM (*Global System for Mobile communication*) com soquete para o cartão SIM (*Subscriber Identification Module*);

b) Desenvolvimento de uma placa com micro controlador para controle do sistema;

c) Aquisição e tratamento dos dados do módulo GPS (*Global Positioning System)*, verificando se já possui uma posição válida e se o veículo está parado ou em movimento;

d) Desenvolvimento de um programa em *Assembly* que faça o gerenciamento do módulo GSM, realizando as conexões necessárias, enviando SMS quando necessário e fazendo o intercâmbio de informações entre os módulos GPS e GSM;

e) Criar rotinas para envio dos dados tratados para o servidor já com estrutura do protocolo HTTP (*Hypertext Transfer Protocol*);

f) Ajuste dos níveis de sinal diferenciados entre os componentes do sistema.

1.2.2.2. SERVIDOR

a) Desenvolvimento de um banco de dados *MySQL* para armazenamento de informações de usuários, veículos, módulos e posições rastreadas;

b) Desenvolvimento de um sistema *PHP (Hypertext Preprocessor)* com comunicação com o banco de dados para realização de cadastro de usuários e veículos;

c) Escuta e recebimento de pacotes HTTP enviados pelo módulo instalado no carro, decodificação das posições GPS envidas no pacote e inserção das mesmas no banco de dados;

d) Interpretação das posições e exibição das mesmas em mapa utilizando *API (Application Programming Interface) Google Maps*.

1.2.2.3. INTERFACE COM O CLIENTE

a) Comunicação do dispositivo eletrônico instalado no veículo com o celular cadastrado do cliente, informando em tempo real em caso de violação do alarme ou movimentação indevida;

b) Desenvolvimento de uma interface gráfica (portal *web*) simples para acesso restrito de cada cliente com funções de gerenciamento e rastreio de seus próprios veículos.

2. FUNDAMENTAÇÃO TEORICA

2.1. GSM E GPRS

2.1.1. GSM

Para a comunicação entre o *hardware* instalado no veículo, o servidor e o telefone celular do usuário, é necessária a utilização de uma rede de comunicação móvel. A escolha da rede GSM foi praticamente automática, já que é atualmente o sistema de comunicação móvel mais utilizado em celulares no mundo, com quase 90% dos assinantes, o que lhe garante também, maior área de cobertura. [4] Além disso, oferece alto grau de liberdade, já que a identificação do usuário está no cartão SIM, que é adquirido separadamente do módulo de comunicação. Assim, o usuário pode contratar os serviços da operadora de sua preferência, sem problemas como a necessidade de alterações ou troca no produto.

Seu espectro de comunicação abrange quatro bandas diferentes no Brasil, sendo elas 850MHz, 900MHz, 1800MHz e 1900MHz e como mantém uma harmonia entre os espectros na maior parte do mundo, o produto poderá ser usado em mais de 218 países diferentes, com o mesmo sistema de comunicação.[5]

Para os avisos enviados aos usuários, será usado o sistema de mensagens de texto padrão da rede GSM, o SMS, que permite o envio de mensagens com até 160 caracteres para o usuário.

2.1.2. GPRS (*General packet radio service)*

A rede GSM foi criada principalmente para telefonia móvel, mas uma serie de serviços agregados também foi definida, incluída a comunicação de dados a até 9600 bps, usando comutação por circuitos. Na época de sua criação, no fim da década de 1980, essa era uma taxa favorável, quando comparada às conexões fixas. Porém, em poucos anos, a taxa de transmissão das conexões fixas cresceu drasticamente. Com a estrutura existente na rede GSM, não era possível desenvolver taxas de transmissão

comparáveis, até que no início da década de 2000 passaram a ser oferecidos os primeiros equipamentos utilizando o sistema GPRS, que permite agregar diversas portadoras para aumentar a velocidade de comunicação e o uso de comutação por pacotes.[6] Potencialmente, a rede GPRS pode alcançar até 171 kbit/s agregando até oito *timeslots* da rede GSM, mas geralmente essa velocidade não é oferecida, já que enquanto um *timeslot* está sendo ocupado, ele não pode ser usado para outras finalidades. Tipicamente, são oferecidas taxas de comunicação de 40 kbit/s para *downlink* e 14 kbit/s para *uplink*.[7]

2.1.3. ESCOLHA DO MÓDULO

Inicialmente, o módulo de comunicação GSM escolhido foi o SIM900, que é homologado pela Agência Nacional de Telecomunicações, ANATEL[8]. Este módulo vem sendo muito utilizado em equipamentos que utilizam GPRS e é o modelo de mais alta tecnologia da SIMCom.[9] Posteriormente, foi trocado para o SIM300C, versão anterior ao SIM900 para mesmo tipo de aplicações, pois houve uma maior facilidade tanto na aquisição do módulo, quanto na aquisição de um kit de desenvolvimento para o mesmo. Além disso, as diferenças entre o funcionamento do SIM300 e do SIM900 são muito pequenas, como maior tolerância de níveis de tensão e algumas mudanças em comandos, por exemplo, “at+cdnsorip”, que passou a ser automático, deixando de existir no novo módulo.[10] Também foi cogitado o uso de um módulo híbrido de GSM e GPS, como o SIM908 do mesmo fabricante, mas essa opção foi descartada para que o GPS pudesse ser colocado como um opcional, caso o cliente requisitasse apenas os avisos de disparo do alarme.

O módulo SIM300C é *tri-band*, funcionando nas bandas de 900MHz, 1800MHz e 1900MHz e possui capacidade de usar GPRS. Sua interface física possui 60 pinos, incluindo uma interface serial que permite comunicação em diversas *baud rates*, que variam de 9600 *bauds* a 115200 *bauds*. Por meio dessa interface, podem ser enviados os comandos para controle do módulo, que funciona com o padrão de comandos AT para módulos GSM e também uma lista de comandos estendidas, que podem ser

utilizados, por exemplo, para um uso simplificado do protocolo TCP/IP, que já vem integrado.[11]

Para fins de estudo do seu funcionamento, foi utilizado um kit de desenvolvimento fornecido pelo próprio fabricante. Esse kit apresenta todos os possíveis interfaceamentos do módulo, sendo bastante interessante para um amplo entendimento do produto.

2.2. GPS

2.2.1. FUNCIONAMENTO

Para fins de localização, a tecnologia escolhida foi o GPS, *Global Positioning System*, o mais preciso existente atualmente. A localização via GPS é feita usando 24 satélites girando a 20200 km de altura ao redor do planeta Terra, de forma que sempre haja ao menos 4 deles acima do plano horizontal de qualquer ponto do planeta, de forma a captar o posicionamento 3D.

Apesar de os satélites oferecerem precisão de até 1 mm na horizontal, por questões de segurança, o padrão utilizado para usuários comuns, chamado de SPS, oferece cerca de 100m de precisão horizontal.[12]

2.2.2. MÓDULO ESCOLHIDO

O módulo GPS escolhido foi o ME-1000RW, mostrado na figura 3, que possui um interfaceamento simplificado, com apenas seis terminais, sendo dois deles de alimentação e os outros quatro para interface serial em nível de tensão RS232 e LVTTL (*Low Voltage Transistor-Transistor Logic)*, possuindo também uma antena acoplada. Ao contrário do SIM300, esse módulo não precisa do envio de comandos para controle, ele envia continuamente uma sequência de sentenças no padrão NMEA 0183, da *National Marine Electronics Association,* do tipo GPGGA, GPGSA, GPRMC, GPVTG e GPGSV uma vez por segundo.[13] No projeto serão usadas apenas as sentenças GPGGA (*Global Positioning System Fix Data)*, pois possuem todas as informações essenciais.[14]

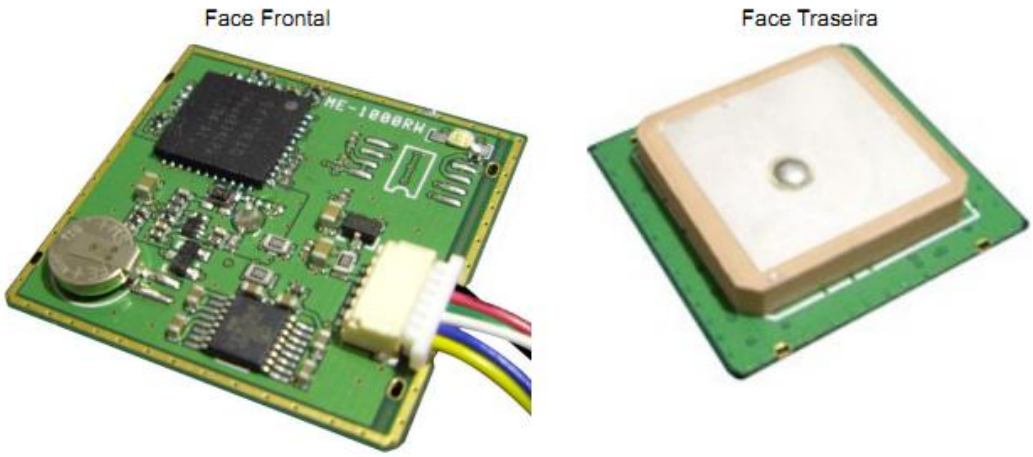


Figura 3 - Módulo GPS 1000RW

2.2.3. O PADRÃO NMEA 0183

É um protocolo que define diversas características para comunicação em 4800 *bauds* de equipamentos de navegação, incluindo o GPS, entre as quais está o formato das sentenças que devem ser enviadas pelo módulo para o sistema. O protocolo também pode ser usado em velocidade de 9600 *bauds*. Para receptores GPS o prefixo das sentenças é sempre “GP”, seguido de uma sequência de três letras que definem o seu conteúdo.[14]

2.3. MICROCONTROLADOR

O microcontrolador (uC) é considerado o cérebro do sistema de *hardware*. Além de possuir uma CPU (*Central Processing Unit)*, ele possui diversos outros elementos como memórias, conversores analógico-digitais e interfaces de comunicação, permitindo que se ligue aos periféricos, controlando os componentes do sistema de forma programada.[15]

O MPC89E515A, da *Megawin*, é um microcontrolador de 8 bits totalmente compatível com padrão da série 80C51, incluindo o conjunto de instruções, porém com melhor desempenho e quantidade de memória. Possui 63 KB de memória *flash* de

programa e mais 1280 bytes de memória RAM, sendo ideal para diversas aplicações. O usuário pode configurar o dispositivo para executar um ciclo de máquina a cada 12 ciclos de *clock* ou, para o dobro de desempenho, a cada 6 ciclos. MPC89E515A tem quatro portas de entrada e saída de 8 bits, três contadores de 16 bits, uma ampla estrutura de interrupções com 6 fontes de interrupção e quatro níveis de prioridade.[16] A escolha foi feita por se tratar de um processador de baixo custo, fácil utilização e por ter grande quantidade de material disponível, já que é uma tecnologia existente a mais de 40 anos.[15] Externamente, será usado um oscilador de cristal de 22,1184MHz para controle de sua freqüência de operação e sua porta serial embutida será usada para comunicação com o módulo GSM. Além disso, uma das demais interrupções será usada para recepção dos dados enviados pelo módulo GPS, também via comunicação serial.

2.3.1. FIRMWARE

O *software* programado que é gravado no uC para controle dos periféricos e tratamento dos dados é chamado de *firmware*. A partir dele que são configuradas as velocidades de comunicação e todo o tratamento dos dados.

O firmware será desenvolvido com utilização da linguagem Assembly, com a lista de comandos projetada para os processadores da família 8051, disponível em seu manual[17].

Para a gravação do firmware no uC, será utilizado um gravador da própria Megawin, modelo TH096A, mostrado na figura 4, específico para gravação desse tipo de processadores, aliado ao seu software de gravação, o 8051 *Writer*, utilizando a opção “EN6T”, que configura um ciclo de máquina a cada 6 ciclos de *clock*.

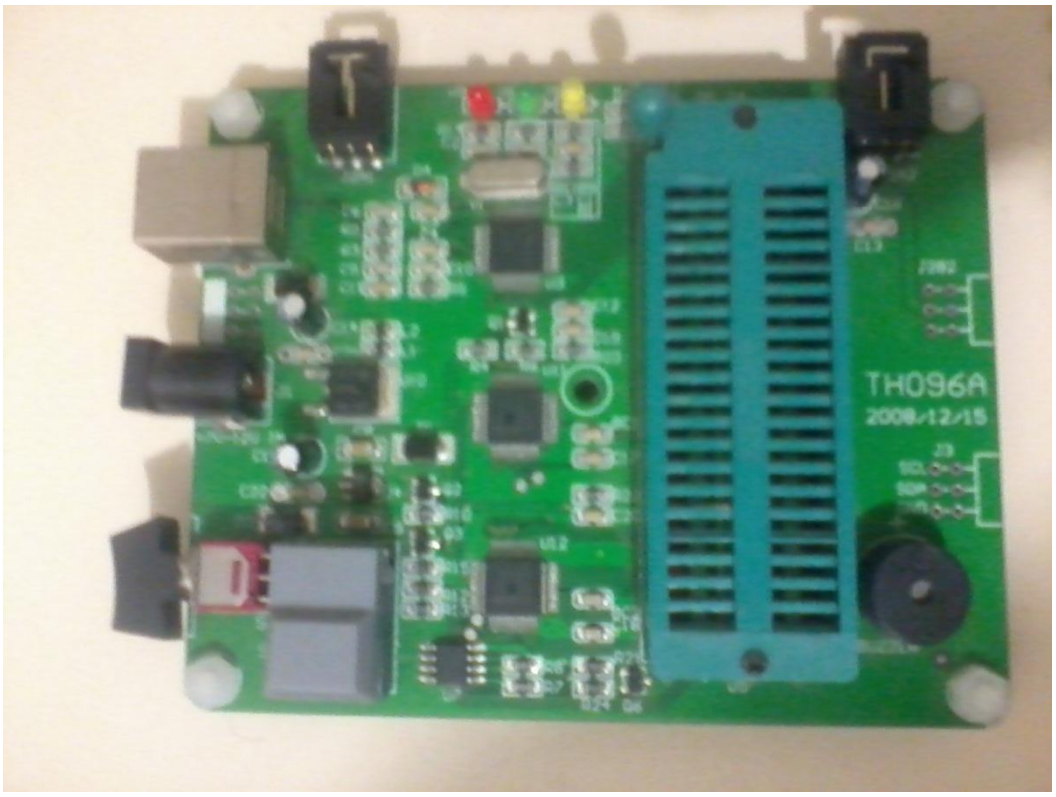


Figura 4 – Gravador TH96A para uCs Megawin.

2.4. COMUNICAÇÃO SERIAL

Enquanto a comunicação paralela se utiliza de oito condutores para entregar um byte de cada vez, a comunicação serial se caracteriza pelo envio de um bit por vez usando apenas um condutor para tal, o que a torna bem mais portátil. Dessa forma, para se obter uma comunicação serial completa bi-direcional, são necessários apenas três terminais, um para envio de dados, chamado de TxD, um para recebimento de dados, chamado de RxD, e um terra comum aos dois. A taxa de envio dos dados é chamada de *baud rate* e medida em *bauds*, que no caso da comunicação serial, é correspondente a bps. A comunicação serial pode ser vista em diferentes padrões e níveis de tensão.[18] Para fins do projeto, serão usados apenas os níveis LVTTL para funcionamento e os níveis do padrão RS-232 para testes em comunicação com um computador.

2.5. I/O NINJA

O software utilizado para testes e verificação da correta comunicação do sistema foi o I/O Ninja, da Tibbo, que permite verificar e realizar comunicação com diversas interfaces do computador. Entre elas, as ferramentas utilizadas serão a leitura e escrita para uma porta serial do computador e a leitura e escrita para uma porta de rede definida no computador, utilizando protocolo TCP.[19]

2.6. PROTOCOLO TCP/IP

O protocolo TCP/IP (*Transmission Control Protocol/Internet Protocol)* define uma série de parâmetros necessários para estabelecimento e controle de conexões entre dois pontos da grande rede.

Cada equipamento em uma rede possui um endereço IP que o identifica nessa rede. Toda a transmissão de pacotes nessa rede leva em consideração esse endereço. O protocolo TCP garante que essa transmissão seja feita de forma confiável, fazendo *checksums* e sequências para os dados transmitidos, para verificar se não há erros, como perda de dados ou desordenação dos pacotes, e reenviar os dados quando necessário.[20]

2.7. SERVIDOR

A definição mais simples de um “servidor” é a de um computador dedicado a rodar um ou mais serviços específicos de forma a suprir necessidades dos demais “clientes” (sejam esses outros serviços ou usuários) de uma rede. Um servidor pode ter diversos papéis, como “servidor de dados”, para armazenamento de arquivos, “servidor de email”, para processamento dos serviços de email e assim por diante[21].

Neste projeto de conclusão de curso, era necessário um servidor “*Web*”, serviço comprometido em hospedar e “fornecer” o conteúdo das páginas *web* pela Internet, assim como um servidor “Banco de dados”, para hospedagem e tratamento dos dados relevantes.

2.7.1. SERVIDOR *WEB APACHE*

O servidor *Web Apache*, desenvolvido pela companhia *Apache Software Foundation* para uma série de sistemas operacionais distintos (*Unix, Solaris, Windows, Mac OS*, dentre outros), trata-se de um *software* de código aberto, ou seja, sua licensa é gratuita e seu código pode ser modificado por outros desenvolvedores de acordo com a necessidade[22]. O Apache é vastamente utilizado em todo o mundo, tendo sido recentemente considerado o servidor *Web* mais popular do mercado[23].

2.7.2. SERVIDOR BANCO DE DADOS *MYSQL*

Para o banco de dados, foi utilizado o *MySQL*, software de "*Structured Query Language*" (Linguagem estruturada de consultas) que roda como servidor provendo acesso aos bancos de dados para múltiplos usuários. O *MySQL* foi desenvolvido pela empresa Sueca MySQL AB (atualmente um *branch* da companhia Oracle) em linguagens C e C++, e, assim como o Apache, é capaz de funcionar em uma série de sistemas operacionais distintos. Diversas linguagens de programação como PHP e *JavaScript*, que também serão utilizados neste projeto, possuem comandos de acesso e comunicação com bancos de dados *MySQL*.

A edição padrão (*Standard Edition*) do *MySQL* é uma plataforma amigável e permite a entrega de aplicações em alta performance e confiabilidade, com simples estruturas para desenvolvimento, *design* e gerenciamento do banco de dados[24], além de demais características que podem ser visualizadas no *datasheet*[25].

2.7.3. EASY PHP

Easy PHP é um software, ou o instalador de um "pacote de softwares" de forma a gerar um ambiente WAMP (*Web development plataform on Windows*, ou plataforma de desenvolvimento Web para Windows) completo para um programador PHP[26]. O ambiente WAMP consiste da combinação do Apache, PHP e *MySQL* para servidores

rodando Windows. A proposta do *Easy* PHP é a de prover este mesmo ambiente em um computador pessoal[27].

Neste trabalho, para testes isolados do servidor do departamento de Elétrica, o Easy PHP foi freqüentemente utilizado.

2.8. PROTOCOLO HTTP

O HTTP, *Hypertext Transfer Protocol*, trata-se de um protocolo de comunicação situado na camada de Aplicação do conhecido modelo OSI, *Open Systems Interconnection model*, modelo de padronização dos elementos de uma rede de comunicação. Uma vez disponíveis os protocolos TCP/IP, previamente abordados neste relatório, o protocolo HTTP é o responsável pelo tratamento de pedidos e respostas entre clientes e servidores na *World Wide Web*.

O HTTP usa a porta 80 para a comunicação entre sítios *Web* e costuma comunicar-se com comandos próprios da sua estrutura, para requisições e respostas, e linguagem HTML para o corpo das mensagens. Uma mensagem, tanto de requisição quanto de resposta, é composta, conforme definido na RFC 2616[28], por uma linha inicial, nenhuma ou mais linhas de cabeçalhos, uma linha em branco obrigatória finalizando o cabeçalho e por fim o corpo da mensagem, opcional em determinados casos.

Neste projeto de conclusão de curso, o protocolo HTTP é utilizado para a comunicação entre o módulo instalado no carro e o servidor. O processador do circuito interpreta as informações envidas pelo GPS e estrutura-as em HTTP, utilizando o comando de envio “POST”. O modulo GSM endereça esta mensagem ao servidor por TCP/IP, que por sua vez recebe os pacotes na porta 80. A partir dai, a página *Web*, em HTML e PHP, irá tratar a informação recebida.

2.9. PROTOCOLO FTP

O protocolo de transferência de arquivo, ou *File Transfer Protocol* (FTP), é um protocolo de rede da camada de Aplicação do Modelo OSI, provendo a funcionalidade de transferência de arquivos entre elementos de uma rede TCP/IP, tal qual a *internet*[29].

Neste trabalho, o protocolo FTP foi utilizado para transferir arquivos para o servidor Web do departamento de Engenharia Elétrica. O software instalado para esta função é o net2ftp e este pôde ser acessado através do endereço "http://lab.eletrica.ufpr.br/webftp/index.php".

2.10. HTML E CSS

HyperText Markup Language, ou HTML, é a linguagem de marcação destinada a programação de páginas na *World Wide Web*. As páginas do HTML são interpretadas do lado do cliente, através do navegador sendo utilizado por este.[30]

A linguagem HTML é escrita em texto simples e cada uma das suas "etiquetas"(ou funções) é precedida e procedida por colchetes ("<" e ">") de forma a delimitar onde começam e terminam. Costumeiramente, uma barra "/" significa o final de uma etiqueta (i.e. <html></html>).

A estrutura básica de uma página HTML se divide em duas partes principais e essenciais, o "cabeçalho" (*head*) e o "corpo" (*body*), como é possível ver na figura 5 a seguir:

Figura 5 - Macro-estrutura de uma página HTML.

As sintaxes <title> e definem, respectivamente, o título da página e uma "divisão" no conteudo. A cada é sugerida a atribuição de um nome especifico.

Para a formatação gráfica (cor de fundo, tamanho da fonte, borda da imagem, etc) de uma série de páginas HTML em um certo padrão, é comum a utilização de CSS (*Cascading Style Sheets*), uma linguagem de estilo que possibilita a separação do conteudo de uma pagina Web da sua formatação gráfica. Para isto, cria-se uma pagina CSS separada onde define-se o estilo desejado de cada divisão. De forma a "empurrar" este estilo dentro do HTML, é utilizada a etiqueta "<link>" para atrelar o arquivo .CSS a cada página pertinente.

Maiores informações tecnicas de linguagem HTML podem ser encontradas em livros texto ou pelo *site.* [31]

2.10.1. HTML Pad

A linguagem HTML é simples e pode ser programada através de um editor de texto comum como o "bloco de notas" do *Windows*. No entanto, para facilitar o desenvolvimento, previa e testes das páginas, é vantajoso a utilização de um *software* dedicado à programação HTML.

Para o desenvolvimento deste projeto foi utilizado o software HTML Pad 2010, editor de diversas linguagens acopladas, como HTML, CSS, Java, dentre outras[31]. O HTML Pad é um software pago, cotado em U$35,85 (trinta e cinco dólares e oitenta e cinco centavos), mas possui uma versão de triagem de 30 (trinta) usos gratuitos.

2.11. PHP

PHP, ou *Hypertext Preprocessor*, é uma linguaguem de programação de "*scripts*" desenvolvida para pruduzir páginas *Web* dinâmicas. O PHP pode ser diretamente inserido dentro do HTML, dispensando a necessidade de realizar "chamada" por outro arquivo para processamento do *script*. Necessariamente, o código PHP será sempre executado do lado do servidor e toda a sua estrutura é transparente para o cliente, que consegue visualizar somente as respostas enviadas. Desta forma, e mediante a sua facilidade de comunicação com bancos de dados, o PHP possibilita a criação de customizações cliente-a-cliente, como uma área de acesso restrito por usuário e senha, além de poupar processamento por parte do computador do cliente, uma vez que todos os scripts são realizados no servidor. Por uma estatistica de 2007, sabe-se que PHP é a linguagem utilizada em mais de 20 (vinte) milhões de *websites* e um milhão de servidores *web*[32].

Assim como o HTML possui as suas "etiquetas" para definição de uma pagina, o PHP tambem possui marcadores de inicio e fim de um código PHP – estes são "<?php" e "?>" para abertura e finalização, respectivamente. Todo o codigo contido dentro destas etiquetas será interpretado do lado do servidor, e qualquer resposta a ser enviada para o cliente virá dentro de aspas após a etiqueta "echo", como é possivel ver no exemplo a seguir:

Em HTML, na pagina inicial, encontra-se um simples formulario HTML que, com a etiqueta "*input*" requisita e envia o nome do usuário. A etiqueta "input" tipo "text" permite entrada de texto, e tipo "submit" executa a ação definida em " form action", a chamada da pagina . Na figura 6 é possivel visualizar a forma como o usuario visualisará este codigo

```
<html>
<head>
<title>Exemplo</title>
</head>
<body>

<form action="acao.php" method="post">
 <p>Seu nome: <input type="text" name="nome" /></p>
 <p><input type="Submit" value="Enviar" /></p>
</form>

</body>
</html>
```

Figura 6 – Exemplo de formulário HTML.

A página “acao.php” trará o seguinte código misto:

```
<html>
<head>
<title>Exemplo</title>
</head>
<body>


Olá <?php echo "htmlspecialchars($_POST['nome'])"; ?>.

</body>
</html>
```

Supondo que o cliente digitasse “João” no campo da pagina inicial e procedesse pelo botão “Enviar”, a página “acao.php” traria somente como resposta “Olá João.” e nada dentro das etiquetas PHP seria visualizavel, mesmo no código fonte, salvo as respostas das consultas - neste caso, “João”.

Consultas mais profundas no assunto podem ser realizadas no manual official PHP. [33] Utilizando esta estrutura básica e conhecendo demais funções PHP, HTML e

CSS, juntamente com auxilio da linguagem JavaScript, o portal da empresa fictícia AAI foi desenvolvido.

2.12. API GOOGLE MAPS

Application Programming Interfaces (ou Interfaces de programação de aplicativos) são conjuntos de rotinas padronizadas por um certo programa e que permitem que outros programas o acessem e utilizem partes das suas funcionalidades. De acordo com o *Computer World*[34], quase todas as aplicações dependem das API's dos sistemas operacionais de forma a realizar certas funções básicas como acesso ao sistema de arquivos. As sintaxes das API's podem ser encontradas nas documentações das aplicações sendo chamadas, e provê-las é uma forma inteligente de garantir que a aplicação dure e interaja beneficamente com outros *softwares*.

Neste projeto, a API da conhecida aplicação *Google Maps*, da companhia homonima, será utilizada de forma a fornecer as posições geograficas do veículo rastreado através de um mapa amigável, assim como geocodificar as posições (latitude e longitude) em endereços.

O funcionamento da API Google Maps baseia-se em cima de código *JavaScript*, com uma série de rotinas e sintaxe padronizada para as chamadas de mapa, conexão com servidores Google e interpretação de posição[35]. As funções, instruções de uso e exemplos de código podem ser encontradas no *website* "*Google Code*".[36]

3. MÉTODOS

3.1. DISPOSITIVO ELETRÔNICO

O circuito do módulo a ser instalado no veículo é dividido em 3 blocos principais, sendo eles uma placa com o módulo GSM, o módulo GPS e uma placa com o microcontrolador.

3.1.1. MÓDULO GSM

Inicialmente foi feito um estudo do funcionamento do módulo SIM300C, com o apoio de um kit de desenvolvimento para produto. O kit, fornecido pelo próprio fabricante, vem preparado para a utilização de todas as interfaces do módulo, diretamente com comunicação com o PC. Portanto, antes da automatização do sistema, foram verificados todos os comandos AT necessários para o envio de SMS, identificação da operadora, identificação do IMEI, conexão GPRS, envio de dados e outras funcionalidades úteis para o sistema. Após essas verificações, partiu para a placa que seria efetivamente utilizada no projeto.

Como a tensão de alimentação do circuito serão os 12V fornecidos pela bateria do veículo e a tensão de alimentação do módulo, VBAT, pode ser de 3,4V a 4,5V, a placa possui um circuito de controle na alimentação, reduzindo sua tensão de entrada. O esquemático desse circuito está mostrado na figura 7. Com a tensão de 12V na entrada do regulador de tensão LM7805, sua saída será de 5V, tensão VCC, com até 1,5A de corrente. Como a tensão VBAT é ainda menor que isso, coloca-se um diodo entre VCC e VBAT, causando uma queda de tensão de aproximadamente 0,7V, para a alimentação do módulo em 4,3V. Além disso, ao ligar o circuito, o LED LD3 irá se acender. A figura 8 apresenta todas as demais ligações existentes na placa. Devido à existência de componentes smd na ligação do módulo GPRS, seria bastante difícil a montagem dessa parte do circuito em uma placa de circuito universal, portanto, foi adquirida uma placa pronta com as características indicadas e os conectores

necessários. A figura 9 mostra as vistas da placa por cima e por baixo. As figuras 10 e 11 mostram o kit de desenvolvimento e o módulo GSM, respectivamente.

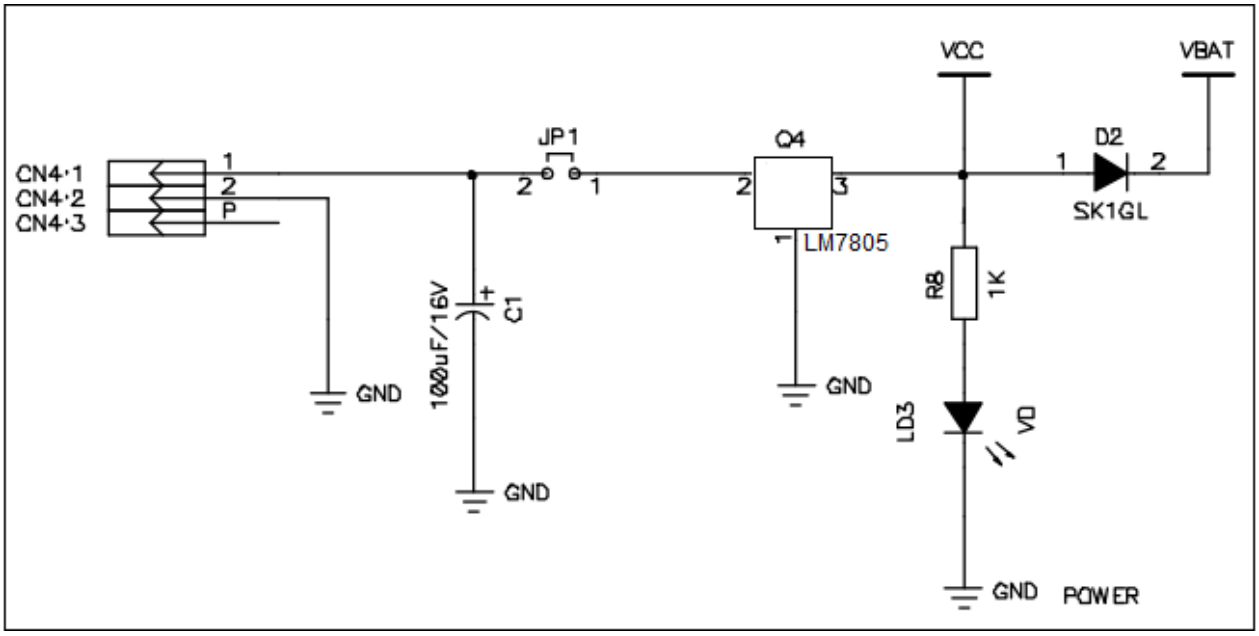


Figura 7 - Circuito de alimentação do módulo GSM.

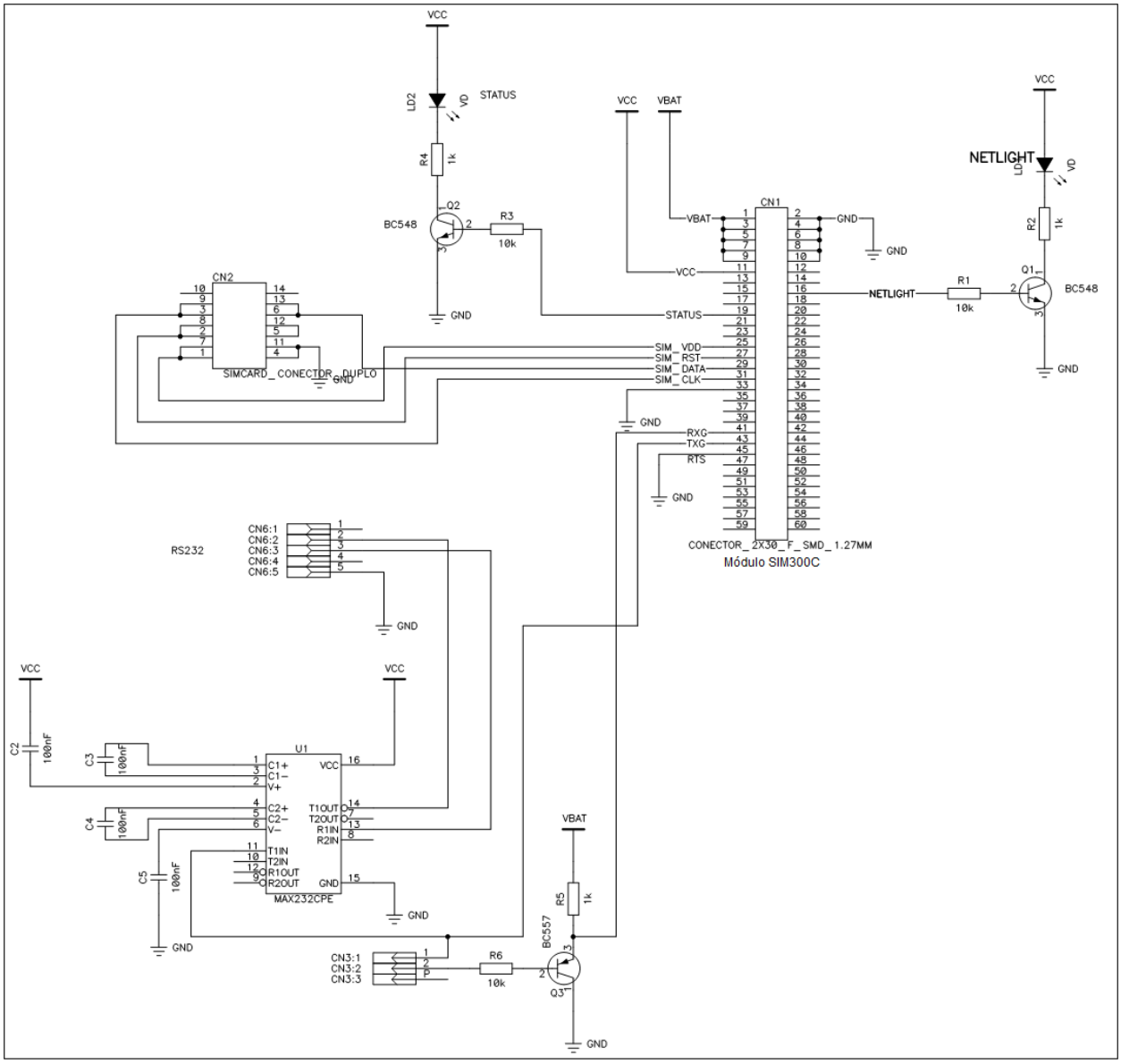


Figura 8 - Circuito da placa do módulo GSM.

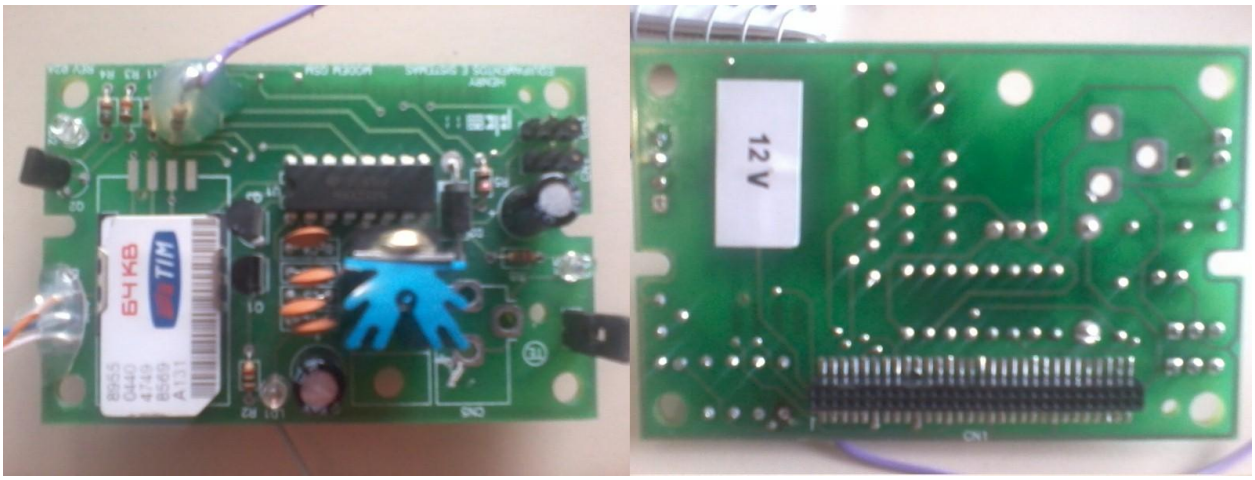


Figura 9 - Fotos da placa de controle do módulo GSM vista de cima e de baixo.

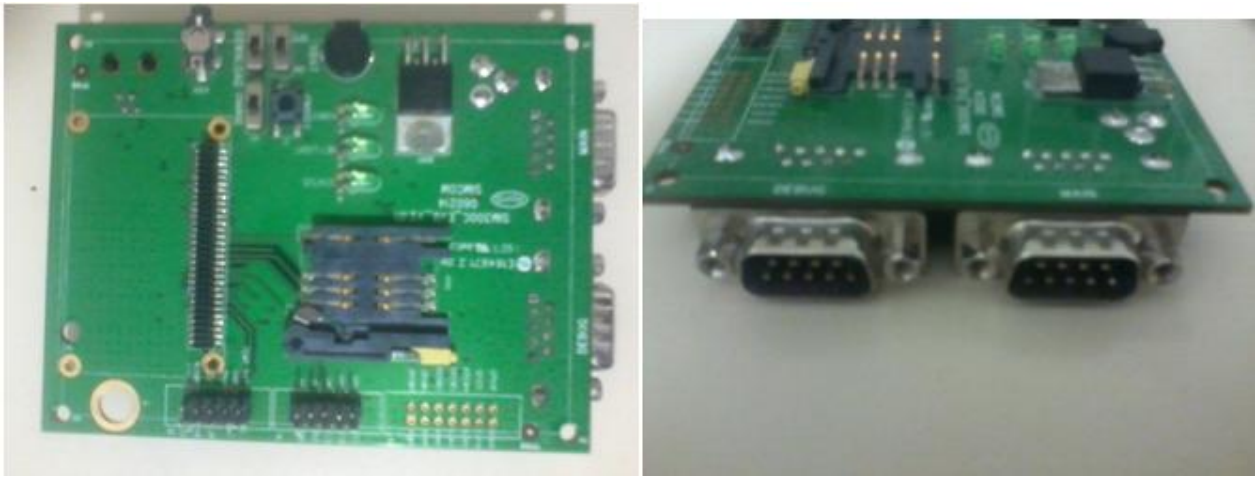


Figura 10 – Fotos do kit de desenvolvimento para SIM300

Figura 11 - Modulo GSM com antena ligada ao pad.

.

Com a configuração da figura 7, sempre que o módulo for ligado, ele estará em modo “carregar bateria” (*ghost mode*), sem poder realizar nenhuma de suas funções. Para colocá-lo em modo de funcionalidade total, é necessário o envio de um comando AT para tal. LEDs também foram colocados nos pinos STATUS e NETLIGHT do módulo. O LED do pino de status irá se acender assim que o módulo for iniciado, mesmo que em modo “carregar bateria” e o netlight começa a piscar após a mudança para o estado de funcionamento total.

O MAX232 é um CI que faz a interface entre portas seriais em nível de tensão RS232 e LVTTL e está presente no circuito apenas para realizar o *sniffer* da porta serial no computador, utilizando o I/O Ninja, mostrado na figura 12. Observa-se a presença de propagandas, pois apesar de ser um programa pago, foi usada a versão de teste. A lista completa de comandos AT utilizados e suas respectivas descrições está no Anexo A.

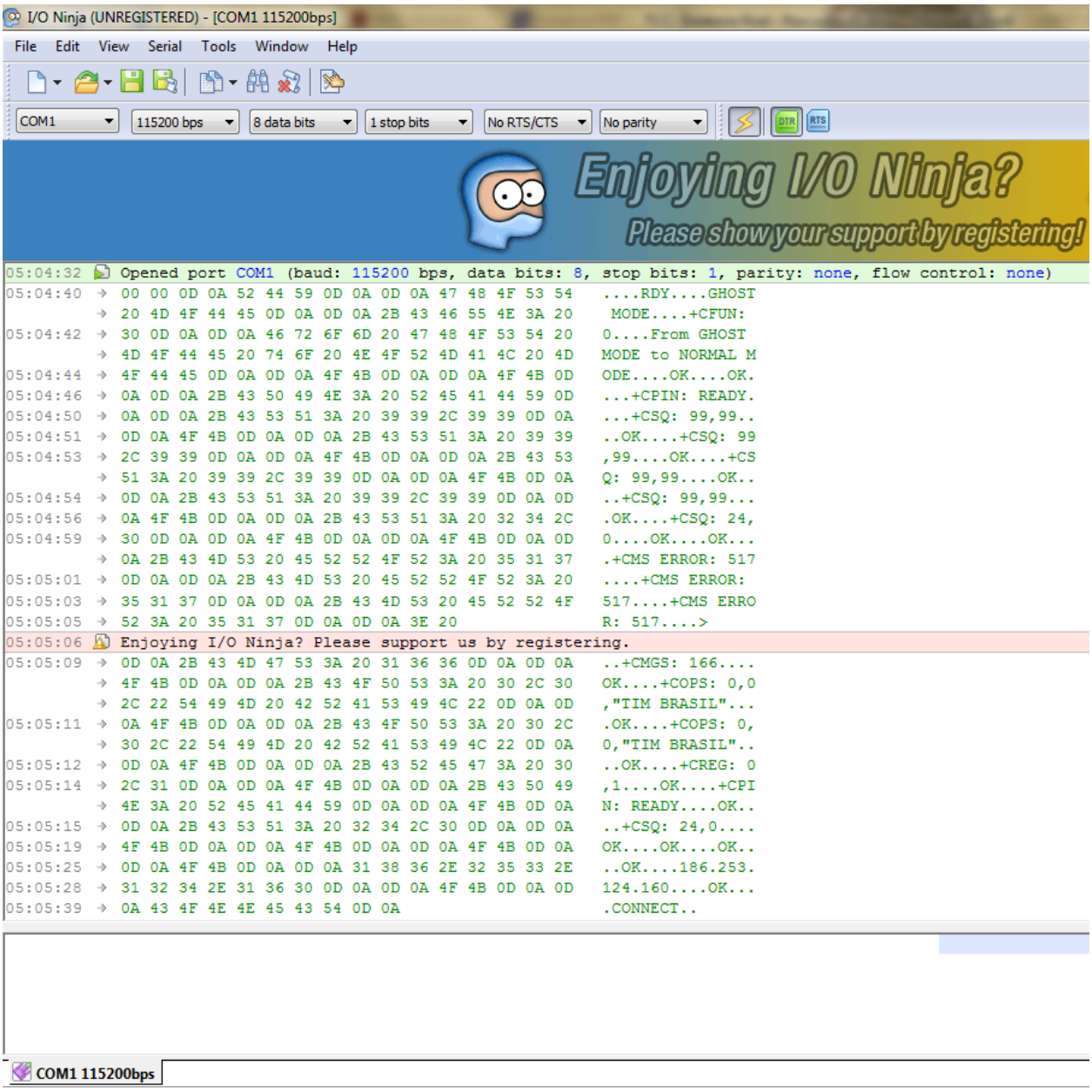


Figura 12 - I/O Ninja com sniffer de uma conexão realizada com sucesso.

Inicialmente a comunicação da placa com o computador era bi-direcional, porém, ao fazer a interface do SIM300C com o uC, observou-se que a excursão de saída do MAX232 era muito forte, impedindo a leitura dos dados enviados por parte do módulo. Por esse motivo, a comunicação com sentido computador-módulo foi cortada, sendo apenas lidos no *sniffer* os dados que o módulo envia para o uC. A porta CN3:1 representa a saída da porta serial do módulo (TxG), que será enviada tanto para o uC

(RxD) quanto para o MAX232 e a CN3:2 representa a entrada para a porta serial do uC (TxD). O transistor entre a porta CN3:2 e o RxG visa o ajuste da tensão de comunicação entre os equipamentos. A porta CN6 representa a porta RS232 que será ligada ao computador, apenas com finalidade de *sniffer* dos dados da comunicação. O pad de antena é inexistente na placa, pois está localizado no próprio módulo GSM.

3.1.2. MÓDULO GPS

O módulo GPS será usado diretamente, tendo seus terminais ligados diretamente à placa de controle, que fará as adaptações de tensões necessárias. Como a comunicação do GPS também é serial e a porta serial do uC já está em uso, foi utilizada uma interrupção externa do uC como serial. Para tal, foi analisado o funcionamento da serial normal e feita uma rotina que funcionasse de forma semelhante. Como a velocidade de comunicação do módulo GPS é 9600 bauds, ou seja, 9600 bits/s, inverteu-se esse valor para chegar ao tempo necessário para o envio de cada bit, que é de 104 us. Com esses valores, foi feita uma rotina que simula uma interrupção serial e logo depois capta os dados enviados.

3.1.3. PLACA PARA GERENCIAMENTO

Para controle total do sistema, foi desenvolvida uma placa constituída basicamente por um uC MPC89E515A, conforme foto na figura 13 e esquemático na figura 14.

Os transistores Q5, Q6, Q7 e Q8 são usados para normalização das tensões entre os equipamentos. O transistor Q9 funciona como chave transistorizada para a alimentação do módulo GPS, para que possa ser mantido desligado enquanto não está em uso. Devido à configuração utilizada, ele possui uma lógica invertida, ou seja, ligará o módulo quando o pino estiver com lógica zero.

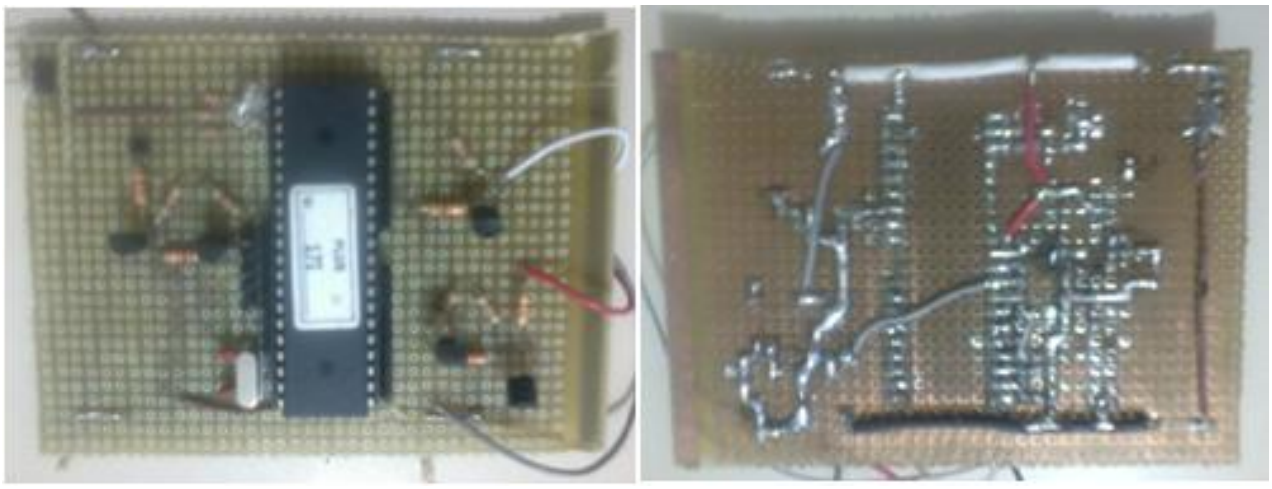


Figura 13 - Fotos da placa para gerenciamento

3.1.4. FIRMWARE

O fluxograma da figura 15 demonstra de forma simplificada o funcionamento do firmware. Por motivos de simplificação, não foram mencionados os *timeouts* existentes nos comandos enviados e alguns dos regimes de correção de erros, que podem tentar reiniciar a conexão ou o modem.

Para cálculo da mudança de posição, verifica-se apenas se o primeiro *byte* depois da vírgula mudou, tanto na latitude, quanto na longitude, o que leva a uma precisão de cerca de 100m. Cogitou-se a utilização do byte seguinte, mas com essa simplicidade de verificação, a diferença seria sempre dez vezes mais ou menos. Como 10m seria uma distância muito curta, considerando a precisão do módulo GPS, considerou-se 100m o valor ideal para utilização.

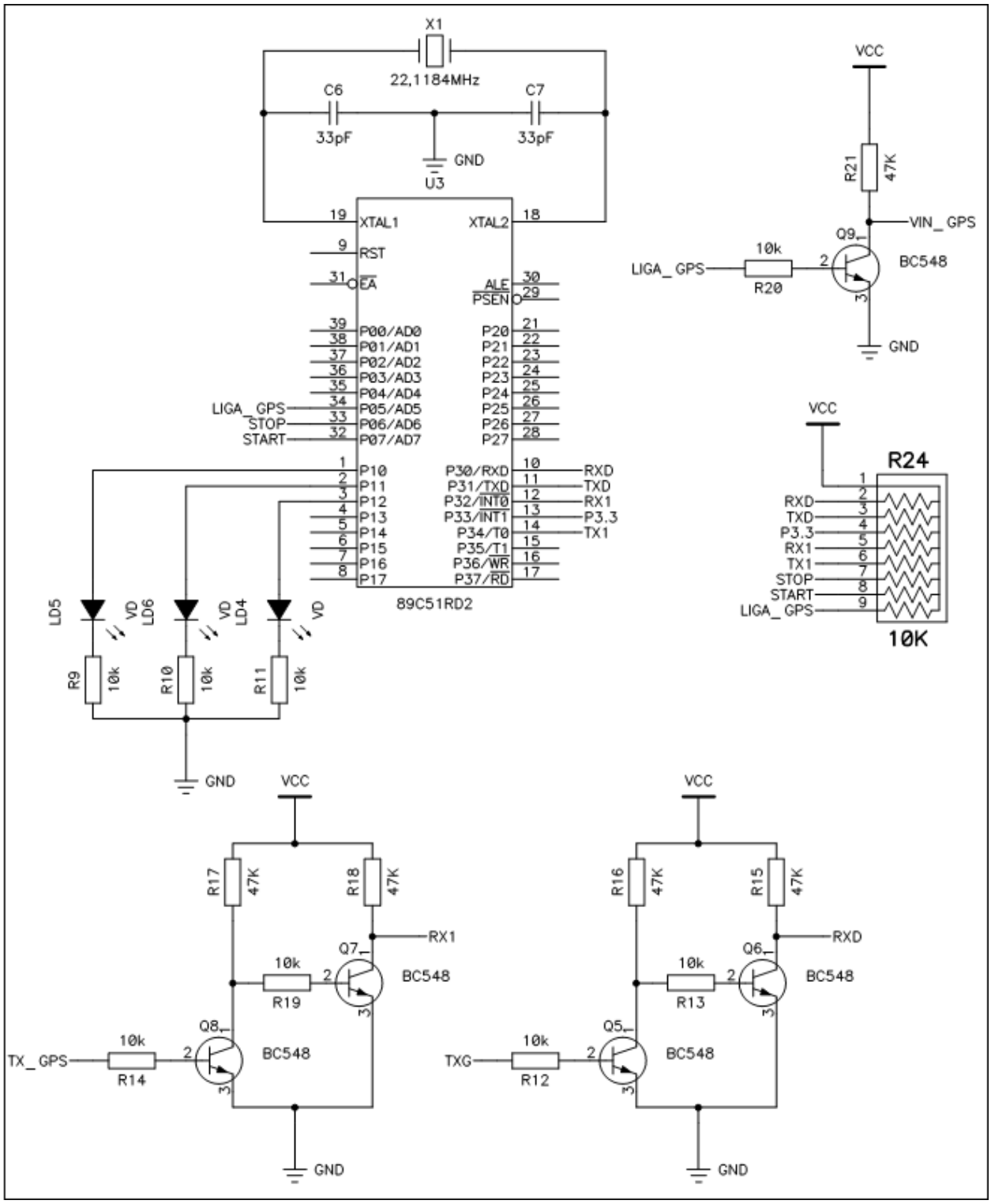


Figura 14 - Esquemático da placa de controle do sistema

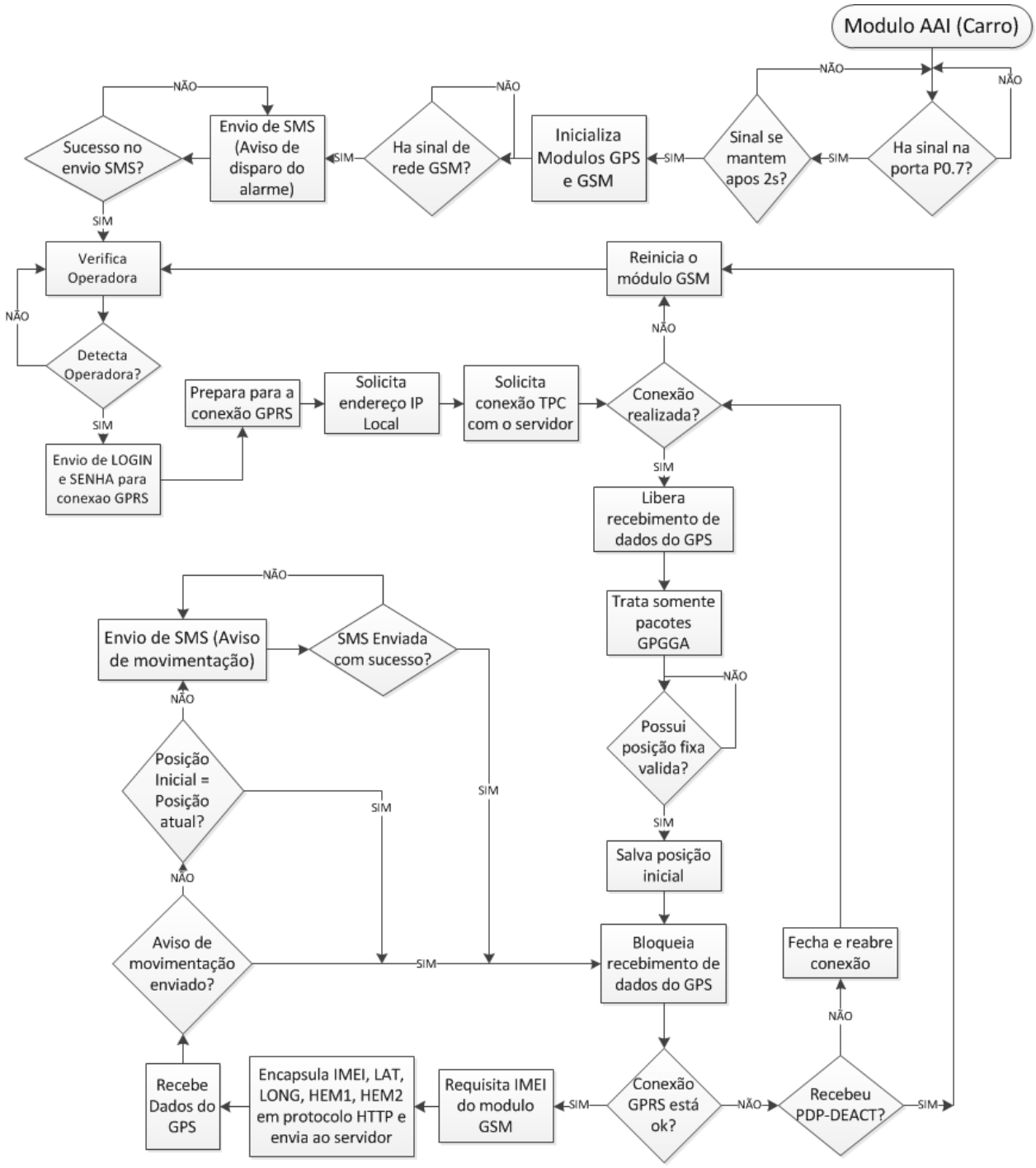

Modulo AAI (Carro)
Sinal se mantem apos 2s?
Ha sinal na porta P0.7?
Inicializa Modulos GPS e GSM
Ha sinal de rede GSM?
Envio de SMS (Aviso de disparo do alarme)
Sucesso no envio SMS?
Verifica Operadora
Detecta Operadora?
Envio de LOGIN e SENHA para conexao GPRS
Prepara para a conexão GPRS
Solicita endereço IP Local
Solicita conexão TPC com o servidor
Conexão realizada?
Reinicia o módulo GSM
Libera recebimento de dados do GPS
Trata somente pacotes GPGGA
Possui posição fixa valida?
Salva posição inicial
Bloqueia recebimento de dados do GPS
Fecha e reabre conexão
Recebeu PDP-DEACT?
Conexão GPRS está ok?
Requisita IMEI do modulo GSM
Encapsula IMEI, LAT, LONG, HEM1, HEM2 em protocolo HTTP e envia ao servidor
Recebe Dados do GPS
Aviso de movimentação enviado?
Posição Inicial = Posição atual?
SMS Enviada com sucesso?
Envio de SMS (Aviso de movimentação)
SIM
NÃO

Figura 15 - Fluxograma representando o funcionamento do firmware do projeto.

3.2. SERVIDOR

O Portal de acesso AAI, ou *website*, foi criado utilizando as linguagens de programação doravante mencionadas e lógicas de programação bem conhecidas. Segue uma explicação simplificada dos métodos e funcionalidades do Portal.

Dentro do banco de dados “erica” foram criadas quatro tabelas a serem usadas ao longo das rotinas do portal AAI. Na figura 16 estão mostradas as tabelas mencionadas, assim como os valores de colunas para cada uma:

DB TABELA “MODULOS”

modulo	telefone	nome	cpf
353655040383071	99098554	Erica Gomes	65076079670

DB TABELA “USUARIOS”

id	nome	usuario	senha	cpf	sexo	telefone
1	Teste	teste	12345	00000000000	mulher	4100000000
2	Erica Gomes	eogomes	12345	07622079650	mulher	4199098554

DB TABELA “VEICULOS”

id	modulo	placa	modelo	cor
2	353655040383072	KVY1313	Peugot	Preto
2	353655040383071	EBR5101	Sandero	Cinza

DB TABELA “POSICAO”

modulo	latitude	longitude	datahora
353655040383071	-25.456586	-49.228127	2011-12-31 11:28:25
353655040383071	-25.454919	-49.228588	2011-12-31 11:29:17

Figura 16 – Tabelas criadas no banco de dados MySQL.

No momento da venda de cada alarme, a empresa fictícia AAI fará uma simples modificação do hardware para cadastro do telefone com o qual o módulo se comunicará e, ao mesmo tempo, irá cadastrar o módulo (numero do IMEI, endereço físico do

módulo GPRS) vendido juntamente com o telefone associado e informações básicas do cliente para consultas futuras.

O cliente da empresa fictícia AAI, havendo realizado a compra e instalação de um módulo em seu veiculo, recebe o manual com o número do módulo e é instruído a entrar no portal AAI e cadastrar-se. Cada usuário cadastrado é único e será associado a um número de identificação (ID) único dentro do sistema. Após realizar cadastro pessoal no portal, o cliente possui um usuário e senha escolhidos por si próprio e com os quais poderá acessar as páginas restritas.

Em seguida, é esperado que o cliente cadastre o(s) veículo(s) que possui. A cada cadastro de veiculo, é necessário ter em mãos o numero do módulo e este não pode ter sido previamente vinculado a outro carro ou não existir na tabela “módulos”. A cada veiculo cadastrado, o sistema automaticamente associa o carro ao número de ID de usuário autenticado na pagina e, desta forma realiza consultas de módulos cadastrados por cliente e exibe esta informação caso requisitado.

Ao consultar os veículos cadastrados, o cliente pode escolher, para cada um deles: Deletar, editar ou rastrear. Deletar apagará as informações daquele veiculo por completo. Editar permitirá que o cliente vincule um módulo previamente cadastrado a outro carro, e rastrear permitirá que ele veja as ultimas (ou todas) posições geográficas do seu veículo em caso de movimentação indevida.

Um fluxograma mais detalhado do Portal AAI pode ser visto na figura 17 em seguida:

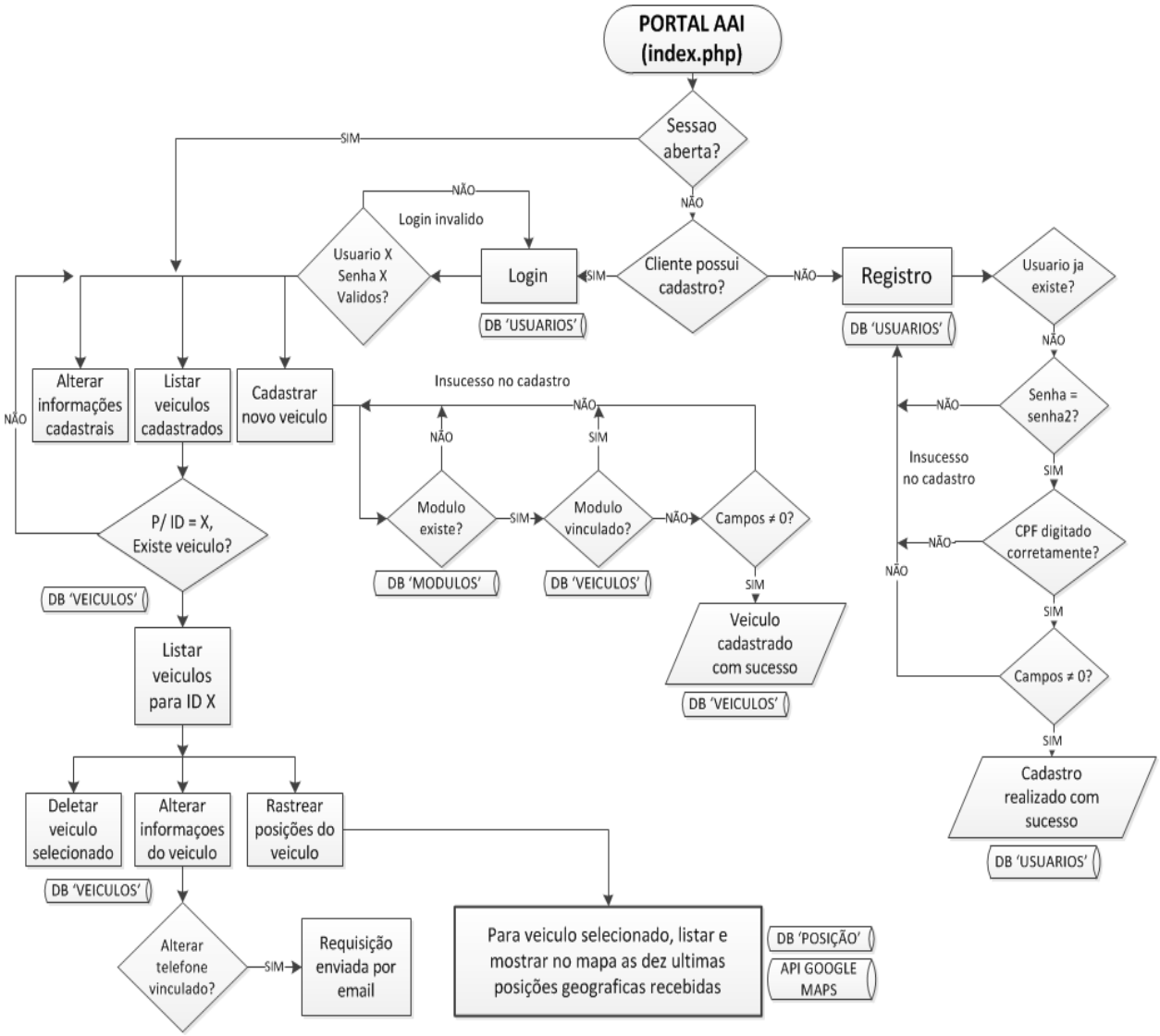


Figura 17 – Fluxograma do portal Web.

Para gerar e manter uma sessão de acesso, foi utilizada uma variável do PHP transicional entre páginas, *"$_SESSION[logado]*", que recebe o valor "1" no momento do primeiro acesso. A cada requisição de pagina seguinte pela parte do cliente, uma rotina inicializa sessão e verifica o valor em *"$_SESSION[logado]*", permitindo que o cliente prossiga caso este valor seja 1. Sendo diferente, a sessão foi desconectada e o cliente é negado acesso em páginas protegidas do *website*.

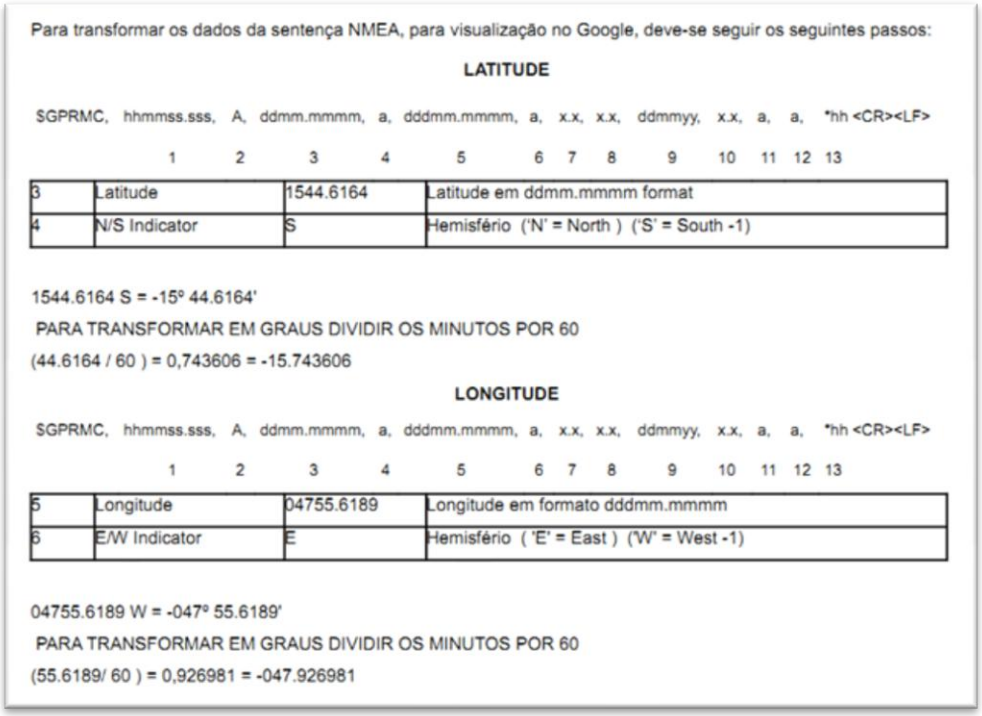
Muitas das rotinas PHP no servidor, como para listagem de veiculos cadastrados ou verificação de um nome de usuário durante o cadastro para evitar duplicadas, são realizadas com “*queries*” (ou consultas) ao banco de dados, por exemplo:

```
$query = "SELECT * FROM veiculos WHERE id='$id'";
$result = mysql_query($query);
$num = mysql_numrows($result);
```

Onde a primeira linha define uma consulta por todos os resultados na tabela “veículos” vinculados a um usuário especifico. A segunda linha envia a consulta ao servidor, e a terceira verifica a quantidade de linhas existem com este resultado. Desta forma, e ao longo de quase todas as funções do site, criamos a comunicação entre o cliente e o banco de dados.

Para o recebimento dos pacotes vindos do GPS, a estrutura é semelhante. O pacote com endereçamento HTTP requisita uma pagina especifica, “recebe_posicao.php”, onde existe uma rotina de entendimento das variáveis recebidas, tratamento dos valores e inserção dos mesmos no banco de dados através de uma “*query*” *INSERT*.

Faz-se necessário tratar os valores recebidos do GPS pois este os manda em formato ddmm.mmmm , como podemos visualizar na Figura 18. No entanto, o *Google Maps* interpreta valores de latitude e longitude em graus.



Para transformar os dados da sentença NMEA, para visualização no Google, deve-se seguir os seguintes passos:

LATITUDE

$GPRMC, hhmmss.sss, A, ddmm.mmmm, a, dddmm.mmmm, a, x.x, x.x, ddmmyy, x.x, a, a, *hh <CR><LF>

1 2 3 4 5 6 7 8 9 10 11 12 13

3	Latitude	1544.6164	Latitude em ddmm.mmmm format
4	N/S Indicator	S	Hemisfério ('N' = North) ('S' = South -1)

1544.6164 S = -15º 44.6164'

PARA TRANSFORMAR EM GRAUS DIVIDIR OS MINUTOS POR 60

(44.6164 / 60) = 0,743606 = -15.743606

LONGITUDE

$GPRMC, hhmmss.sss, A, ddmm.mmmm, a, dddmm.mmmm, a, x.x, x.x, ddmmyy, x.x, a, a, *hh <CR><LF>

1 2 3 4 5 6 7 8 9 10 11 12 13

5	Longitude	04755.6189	Longitude em formato dddmm.mmmm
6	E/W Indicator	E	Hemisfério ('E' = East) ('W' = West -1)

04755.6189 W = -047º 55.6189'

PARA TRANSFORMAR EM GRAUS DIVIDIR OS MINUTOS POR 60

(55.6189/ 60) = 0,926981 = -047.926981

Figura 18 – Pagina 14 do manual do modulo GPS ME1000-RW (ANEXO 2)

Portanto, é realizada a seguinte rotina de conversão:

```
$latitude = $latitude/100;

$latitude = ((($latitude - intval("$latitude"))/60)*100)+
intval("$latitude");

if ($hem1 = "S"){
   $latitude = ($latitude)*(-1);
 }
```

Para apresentar os pontos no mapa e realizar a Geocodificação reversa da *API Google Maps*, foram utilizados comandos *JavaScript* sugeridos pela Google. A Geocodificação reversa trata-se de uma rotina para entendimento de pontos geográficos (latitude e longitude) em endereços amigáveis e mais facilmente entendíveis – País, estado, cidade e CEP.

4. MATERIAIS

4.1. CIRCUITO

Para o desenvolvimento do circuito, foram utilizados:

- 1 módulo GSM SIM300C;
- 1 módulo GPS ME-1000RW;
- 1 uC Megawin MPC89E515A;
- 1 rede resistiva de 10kohm;
- 11 resistores de 10kohm;
- 5 resistores de 47kohm;
- 1 resistor de 4,7kohm;
- 4 resistores de 1kohm;
- 7 transistores BC548;
- 1 transistor BC557;
- 6 LEDs de alto brilho;
- 1 regulador de tensão LM7805 com dissipador de calor;
- 1 cristal de 22,1184MHz;
- 2 capacitores cerâmicos de 33pF;
- 2 capacitores eletrolíticos de 100uF;
- 1 MAX232;
- 5 capacitores cerâmicos de 100nF;
- 1 slot para cartão SIM;
- 1 soquete de 60 pinos para o SIM300C;
- 1 soquete de 40 pinos para o uC.

4.2. SERVIDOR

Para o desenvolvimento do sistema do lado do Servidor, foram utilizados:

- Servidor *Web Apach*e do Departamento de Engenharia Elétrica da UFPR (IP 200.17.220.150);

- Bando de dados *MySQL* do Departamento de Engenharia Elétrica da UFPR (DB “erica”);
- Micro computador pessoal com acesso ao servidor e banco de dados por FTP (Porta 21);
- *Software Easy PHP 5.3.8.1*; - *Software HTML Pad 2010* para programação dos códigos;
- *Software Photoshop CS5 Extended* para desenvolvimento de imagens/ícones.

5. ANALISE DOS RESULTADOS

5.1. HARDWARE

Os diversos testes realizados partiram das partes mais isoladas para as maiores interações do projeto, alguns com resultados esperados, outros adversos. Os principais problemas encontrados foram nas interfaces. Os primeiros testes foram realizados interfaceando o uC com o SIM300C para envio de SMS conexão GPRS com uma listener socket aberta em um computador qualquer. Apesar dos problemas iniciais de comunicação, a retirada do pino do MAX232 do circuito resolveu o problema com certa facilidade e a comunicação, envio de SMS e conexão foram corretos com envio e recebimento de dados.

Ao efetuar os mesmos testes conectando ao servidor do projeto, a conexão é efetuada com sucesso, porém o envio de dados não recebe resposta alguma do servidor, além de não ser entendido pelo mesmo. Além disso, algumas características da comunicação por protocolo HTTP também se mantiverem, como o fechamento externo da conexão após o envio do pacote, mas não foi possível assegurar o recebimento ou não do dos pacotes por parte do servidor, já que não foi encontrado nenhum programa que pudesse ser usado como sniffer da porta utilizada (80).

Posteriormente, observou-se que há um redirecionamento de DNS no servidor do departamento de engenharia elétrica, já que a partir do mesmo IP podem ser acessados, tanto a URL eng.eletrica.ufpr.br quanto URL eletrica.ufpr.br. Partindo disso, novos testes foram realizados, desta vez conectando o módulo GPRS em modo DNS em vez de modo IP. Os resultados obtidos foram os mesmos. Foram feitos testes tanto com conexão em modo transparente quanto em modo normal, os resultados obtidos foram diferentes, mas em ambos os casos sem resposta do servidor. Isso posto, cogitou-se ainda a possibilidade de bloqueio na comunicação por parte da operadora, mas a informação não pode ser confirmada.

Com relação aos dados do GPS, concluiu-se primeiramente que seria melhor o uso de um GPS que funcionasse por meio de comandos, isso evitaria o excesso de

dados na porta serial e a necessidade de acionar e bloquear a interrupção para a aquisição de dados confiáveis. Além disso, mesmo tendo as rotinas prontas, não foi possível fazer de fato o tratamento dos dados do GPS, pois a portal serial virtual não teve variações significativas de tensão, para que os dados fossem captados pelo uC e tratados. Diversas modificações no tratamento da porta foram testadas sem sucesso também. Esforços vêm sendo feitos no sentido de isolar o problema, de forma que possa ser resolvido, ou contornado por alguma solução eficaz.

Por outro lado, todas as outras etapas foram concluídas com sucesso. Após o disparo do alarme, o módulo é ligado e já começa a testar a qualidade do sinal de rede. Assim que o sistema encontra sinal, mesmo que baixo, ele parte para o envio da mensagem SMS, conseguindo na maioria das vezes completar essa etapa em menos de 30 segundos após o disparo do alarme. Logo que recebe confirmação, o módulo parte para a identificação da operadora em que o cartão SIM está conectado, como o sinal de rede já foi detectado, essa etapa se resolve em apenas uma ou duas tentativas, permitindo que o uC escolha automaticamente a APN em que deverá se conectar. Partindo daí, o uC envia para o módulo todos os comandos necessários para preparação do contexto para conexão e finalmente o IP e porta do servidor para conexão, completando o processo todo em cerca de 1 minuto. Após a realização da conexão, foi testado o envio dos pacotes HTTP simulados e foram enviados várias vezes sem erros, sempre com verificação da estabilidade da conexão e correção em caso de necessidade.

5.1.1. ANÁLISE DE CUSTO

O custo final do Alarme Automotivo Inteligente, como esperado durante os estudos preliminares deste projeto, não ultrapassou o valor total de R$205,00 (duzentos e cinco reais), a preço de varejo.

Adicional a isto, o cliente ainda precisaria contratar um plano da operadora de sua preferência e acoplar o chip GSM ao módulo para permitir o envio de mensagens.

No entanto, este alarme, diferente de muitos outros no mercado, não terá custos mensais agregados, pois não haverá uma empresa de segurança dedicada monitorando os veículos.

Na Tabela 1 a seguir estão listados os preços de cada componente utilizado no hardware, sua quantidade e custo final.

Componente	Custo unitario	U	Custo final
1 módulo GSM SIM300C	R$ 80.00	1	R$ 80.00
1 módulo GPS ME 1000RW	R$ 99.00	1	R$ 99.00
1 uC Megawin MPC89E515A	R$ 3.00	1	R$ 3.00
1 rede resistiva de 10kohm	R$ 0.50	1	R$ 0.50
11 resistores de 10kohm	R$ 0.03	11	R$ 0.33
5 resistores de 47kohm	R$ 0.03	5	R$ 0.15
1 resistor de 4,7kohm	R$ 0.03	1	R$ 0.03
4 resistores de 1kohm	R$ 0.03	4	R$ 0.12
7 transistores BC548	R$ 0.11	7	R$ 0.77
1 transistor BC557	R$ 0.10	1	R$ 0.10
6 LEDs de alto brilho	R$ 0.40	6	R$ 2.40
1 regulador de tensão LM7805 com dissipador de calor	R$ 2.00	1	R$ 2.00
1 cristal de 22,1184MHz	R$ 2.00	1	R$ 2.00
2 capacitores cerâmicos de 33pF	R$ 0.10	2	R$ 0.20
2 capacitores eletrolíticos de 100uF	R$ 0.25	2	R$ 0.50
1 MAX232	R$ 1.95	1	R$ 1.95
5 capacitores cerâmicos de 100nF	R$ 0.10	5	R$ 0.50
1 Placa Universal	R$ 10.00	1	R$ 10.00
1 slot para cartão SIM	R$ 1.00	1	R$ 1.00
1 soquete de 60 pinos para o SIM300C	R$ 0.50	1	R$ 0.50
1 soquete de 40 pinos para o uC	R$ 0.10	1	R$ 0.10
		TOTAL	R$ 205.15

Tabela 1 – Custo total do módulo do alarme AAI

Na Tabela 2 a seguir, podemos verificar uma comparação de funcionalidades entre o alarme AAI e demais produtos similares presentes no mercado atualmente, todos utilizando tecnologias GPS e GSM e sem custos agregados de mensalidade.

	AAI	Rastreador e Bloqueador Vias 900-GPS da Svias[37]	Rastreador VTR200[38]	KX500 Rastreador e Bloqueador Automotivo[39]	Furuta Locktronic RLK[40]	XSAT PRO III[41]	Rastreador GPS GT01[42]
Aviso de disparo do alarme por SMS	✓	✓		✓	✓	✓	✓
Aviso de movimentação do veiculo por SMS	✓			✓	✓	✓	✓
Envio da posição do veiculo por SMS					✓		✓
Envio de informação de velocidade por SMS						✓	✓
Bloquear combustível através de SMS		✓	✓	✓	✓	✓	
Ativar sirenes através de SMS		✓	✓	Opcional			
Identificar posição do veiculo pela WEB	✓	✓	Opcional - Serviço pago à parte	✓		✓	
Verificar todas as posições percorridas pela WEB	✓		Opcional - Serviço pago à parte	✓		✓	
Escuta interna embutida		✓		✓		✓	
Aviso de violação da cerca eletrônica					✓		✓
Botão de pânico		✓	✓				
Solicitação de *status* do veiculo por SMS		✓	✓		✓		
Bateria *Backup*		✓	✓	✓	✓		✓
Preço		R$625,00	R$699,00	R%849,00	R$699,00	R$900,00	R$170,00

Tabela 2 – Comparação de funcionalidades do módulo AAI com os concorrentes

Considerando que a inclusão de funcionalidades como envio da posição por SMS, envio de informação de velocidade e aviso de violação da cerca eletrônica (área delimitada) tratam-se apenas da modificação de poucas linhas de código, sem aumento de custo, é possível comparar e até superar o preço do produto mais barato encontrado, igualando funcionalidades.

Em média, o preço de produtos similares, embora estes tenham algumas funcionalidades a mais, gira em torno de R$600,00 (seiscentos reais), o que torna o preço do Alarme AAI bastante competitivo no mercado.

5.2. WEBSITE

O desenvolvimento e estruturação do portal AAI foi satisfatório, capaz de atender o escopo inicial do projeto. As linguagens de programação e interfaces escolhidas foram de fácil aprendizado e utilização, com amplas referencias bibliográficas distribuídas pela *internet* para estudo e consulta. Algumas rotinas, como o sistema de *login* e a geocodificação reversa da *API Google Maps*, requeriam conhecimento mais estruturado e mostraram-se mais custosas de programar com sucesso.

O portal criado, simples em sua estrutura e de fácil entendimento, possibilita ao cliente da empresa fictícia AAI fácil navegação e acesso às informações relevantes do seu carro. As figuras 19-24 a seguir apresentam o resultado final de algumas das páginas do *website*.

Figura 19 – Pagina inicial do portal AAI.

Figura 20 – Página de cadastro de novo usuário.

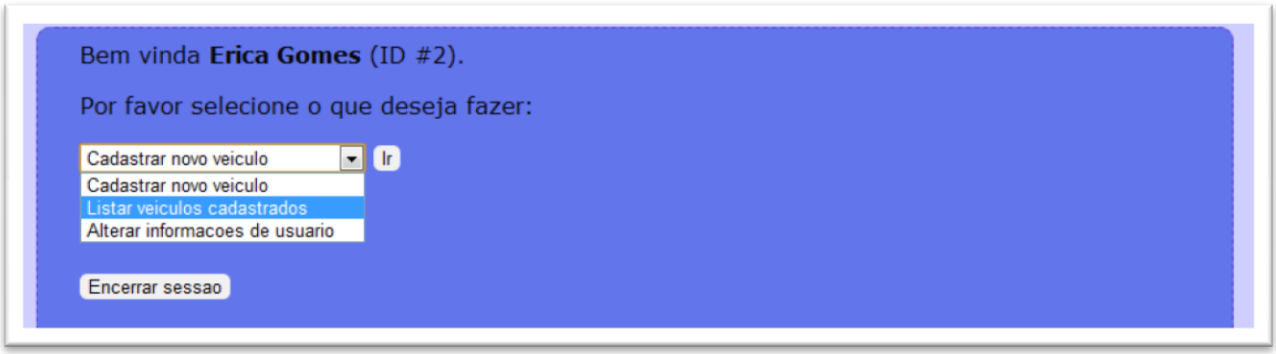


Figura 21 – Página de acesso restrito ao usuário autenticado.

Figura 22 – Listagem de carros cadastrados de um usuário.

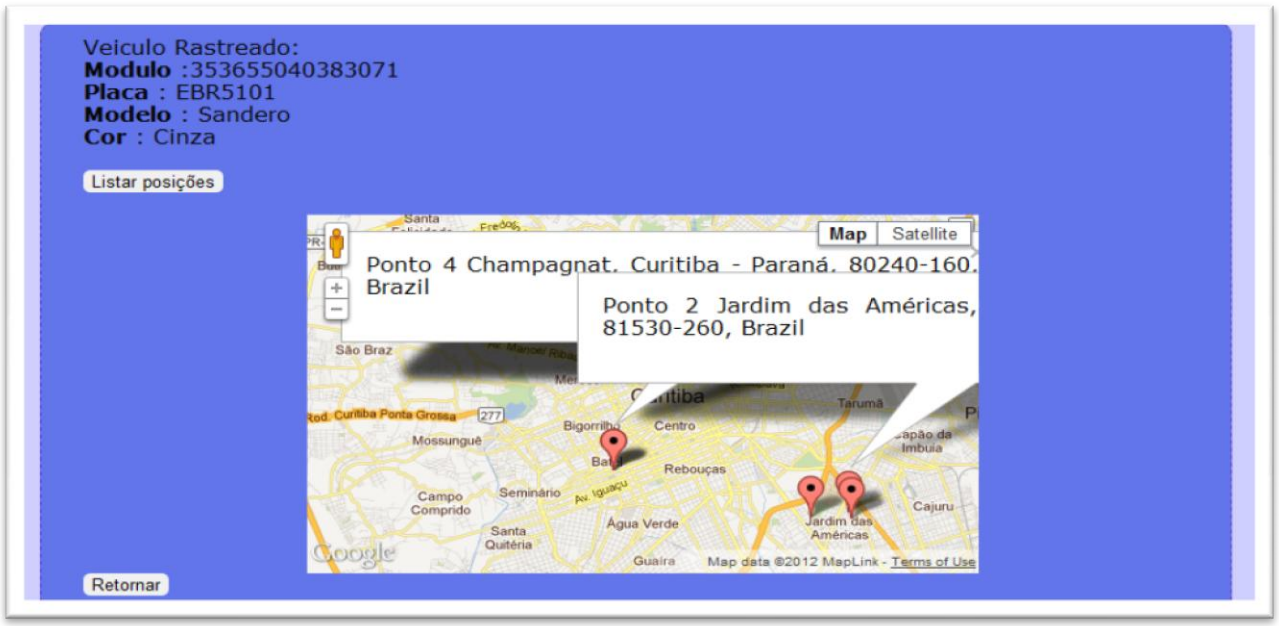


Figura 23 – Página de rastreamento de posições com pontos no mapa.

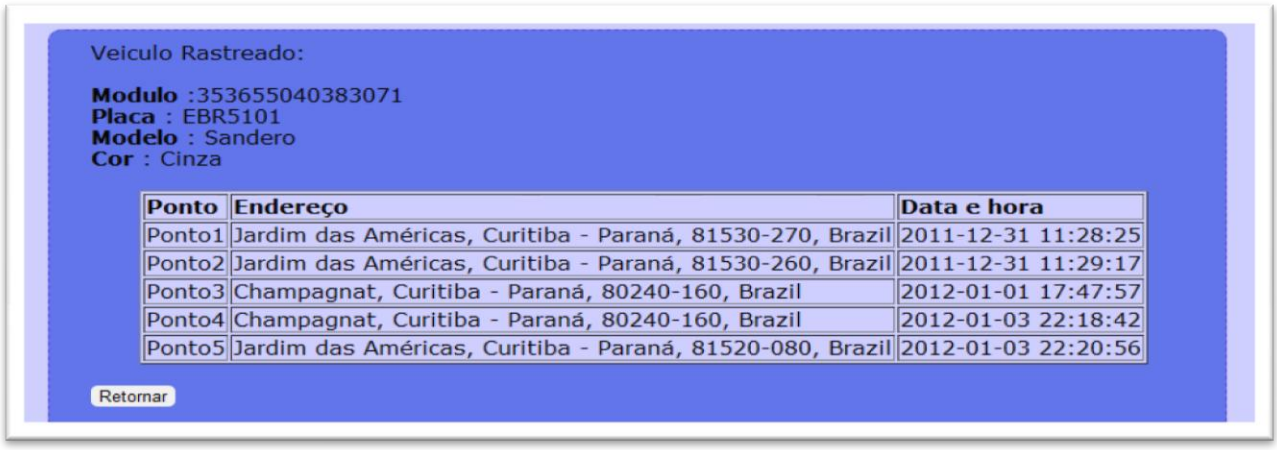

Veiculo Rastreado:
Modulo :353655040383071
Placa : EBR5101
Modelo : Sandero
Cor : Cinza
Ponto | Endereço | Data e hora
Ponto1 | Jardim das Américas, Curitiba - Paraná, 81530-270, Brazil | 2011-12-31 11:28:25
Ponto2 | Jardim das Américas, Curitiba - Paraná, 81530-260, Brazil | 2011-12-31 11:29:17
Ponto3 | Champagnat, Curitiba - Paraná, 80240-160, Brazil | 2012-01-01 17:47:57
Ponto4 | Champagnat, Curitiba - Paraná, 80240-160, Brazil | 2012-01-03 22:18:42
Ponto5 | Jardim das Américas, Curitiba - Paraná, 81520-080, Brazil | 2012-01-03 22:20:56
Retornar

Figura 24 – Página de rastreamento de posição com listagem simples.

Acessando pelo navegador *Microsoft Internet Explorer 8*, o portal perde certas configurações de estilo e requer maior tempo de resposta, não demorando mais do dois segundos para abrir as páginas de mapa e tabela. Pelo *Google Chrome*, navegador utilizado durante o desenvolvimento, as respostas são recebidas em menos de um segundo, o que caracteriza um *website* leve e códigos de rápido processamento.

Ao total, foram programadas 28 páginas para o portal, somando 34Mb de espaço ocupado, desconsiderando os gráficos. Considerando as imagens, todo o *website* utiliza menos de 1.5Gb no servidor.

As quatro tabelas do banco de dados, totalizadas, somam 6Kb de espaço utilizado. Considerando que cada coluna, em média, ocupa somente 600B de espaço e levando em consideração que a tabela mais massiva em quantidade de dados, a tabela “posições”, só será preenchida em caso de roubo do automóvel, pode-se estimar um crescimento baixo e controlável para o banco de dados, não requerendo grande capacidade de disco instalada.

Um servidor de bom desempenho para pequenas empresas, da série *Power Edge* T da Dell, pode ser adquirido a partir de R$1.599,00 (um mil quinhentos e noventa e nove reais)[43]. Considerando que as edições utilizadas dos *softwares Apache* e

MySQL são ambas gratuitas, concluímos que o sistema desenvolvido, da parte de prestação de serviços de Tecnologia da Informação, é de baixo custo capital e requer pouca manutenção.

6. CONCLUSÃO

Buscou-se com este trabalho de conclusão de curso a concepção teórica e prática de um sistema de alarme automotivo inteligente, contemplando *hardware* e *software*. Infelizmente, os resultados práticos finais não atenderam completamente ao escopo inicial.

A comunicação GSM foi obtida com sucesso e o envio de mensagem em caso de violação não apresentou problemas. Muitas dificuldades foram encontradas nos testes e confecção de interfaces, agradas pelo pouco tempo disponível para aprofundar na resolução dos problemas. No entanto, muitas etapas isoladas desenvolvidas e testadas com sucesso.

O software desenvolvido atendeu ao escopo proposto, simples e de fácil utilização por parte do cliente. Com relação à proposta de um “alarme com potencial comercial”, foi possível provar que este produto de fato poderia ser desenvolvido com custos bem mais baixos que os competidores atualmente comercializados no Brasil, tornando este o principal diferencial do projeto.

6.1. MELHORIAS FUTURAS

(a) Comunicação bi-direcional: Tendo em vista que, ao se abrir a porta de comunicação, o envio de dados é bi-direcional, pode-se utilizar esse canal para enviar comandos do servidor para o módulo, de forma a criar novas funcionalidades no sistema:

- Alteração automatica do número de telefone vinculado atraves de formulario disponivel no portal;
- Inclusão de diversos numeros de telefones diferentes a servem vinculados ao módulo;
- Controle de equipamentos dentro do veículo (i.e. ignição) pelo servidor ou pelo celular.

(b) Identificação dos sensores: Sabendo que a maior parte dos pinos do uC ficaram disponiveis, é possivel pensar em implementa-las para criar uma rotina de identificação de qual das portas do carro (qual dos sensores do alarme) foi violado.

(c) Camêra imbutida: Embora a rede GSM não seja a mais propicia para esta funcionalidade, a implementação de uma mini câmera imbutida é possivel e util. Uma única foto do motorista em caso de movimentação indevida do automovel seria suficiente para identificar o ladrão, mesmo que houvessem atrasos no envio.

7. REFERENCIAS

[1]http://oica.net/category/production-statistics/, *The International Organization of Motor Vehicles Manufactures,* ***"Production Statistics"***, 2010, acesso em 31 de dezembro de 2011.

[2]http://www.transalt.org/files/newsroom/reports/caralarms/report.pdf, *Transportation Alternatives*, ***"Alarmingly Useless: The Case for Banning Car Alarms in New York City"***, 21 de março de 2003, pag. 9, aceso em 2 de dezembro de 2011.

[3]http://www.familyhomesecurity.com/, **The Most Reliable Anti-Theft Devices For Cars**, acesso em 4 de dezembro de 2011.

[4]http://www.zdnet.co.uk/news/mobile-devices/2007/09/07/happy-20th-birthday-gsm-39289154/, ZDNet, ***Happy 20th Birthday, GSM***, acesso em 28 de dezembro de 2011.

[5]http://www.gsmworld.com/technology/gsm/index.htm, GSM World, **GSM** acesso em 28 de dezembro de 2011.

[6]http://www.etsi.org/WebSite/Technologies/gsm.aspx, ETSI, ***Mobile Technologies GSM***, acesso em 28 de dezembro de 2011.

[7]http://www.etsi.org/WebSite/Technologies/gprs.aspx, ETSI, ***General Packet Radio Service, GPRS***, acesso em 28 de dezembro de 2011.

[8]http://sistemas.anatel.gov.br/sgch/HistoricoCertificado/Homologacao.asp?NumRFGCT=167611&idtHistoricoCert=47205, ANATEL, **Certificado de Homologação**, acesso em 28 de dezembro de 2011.

[9]http://wm.sim.com/Sim/FrontShow_en/wireless/list_gsm_gprs2.aspx, SIMCom, ***Wireless Module*****,** acesso em 28 de dezembro de 2011.

[10]http://wm.sim.com/Sim/News/photo/2010721162012.pdf, SIMCom, ***SIM900D SIM300D Comparison***, acesso em 28 de dezembro de 2011.

[11]http://www.simcom.us/act_admin/supportfile/SIM300C_HD_V2.04.pdf, SIMCom, ***SIM300C Hardware Manual***, acesso em 28 de dezembro de 2011.

[12]http://www.rc.unesp.br/igce/aplicada/DIDATICOS/LANDIM/textogps.pdf, BERNARDI, José e LANDIM, Paulo, **Aplicação do Sistema de Posicionamento Global (GPS) na Coleta de Dados**, acesso em 2 de janeiro de 2011.

[13]http://www.mecomp.com.br/rumo/ME-1000RW.pdf, MEComp, **Módulo GPS com Antena Acoplada**, acesso em 2 de janeiro de 2011.

[14]http://www.gpsinformation.org/dale/nmea.htm, GPS Information, ***NMEA Data***, acesso em 2 de janeiro de 2011.

[15]http://www.das.ufsc.br/~werner/eel7030/8051/Apostila8051Hari.pdf, HARI, Werner, **A família de microcontroladores 8051**, acesso em 2 de janeiro de 2011.

[16]http://www.keil.com/dd/docs/datashts/megawin/mpc89e515a.pdf, Megawin, **MPC89E515A 8-bit microcontroller**, acesso em 2 de janeiro de 2011.

[17]http://www.atmel.com/dyn/resources/prod_documents/doc4316.pdf, Atmel, ***Atmel 8051 Microcontrollers Hardware Manual***, acesso em 2 de janeiro de 2011.

[18]http://www.taltech.com/support/entry/serial_intro, TALTech, ***Introduction to Serial Communications***, acesso em 2 de janeiro de 2011.

[19]http://tibbo.com/ninja/, Tibbo, **I/O Ninja.** acesso em 2 de janeiro de 2011.

[20]http://www.itprc.com/tcpipfaq/, ITPRC.COM, ***TCP/IP Frequent Asked Questions***, acesso em 2 de janeiro de 2011.

[21]http://technet.microsoft.com/en-us/windowsserver/bb250589, Microsoft, **Microsoft Windows Server**, acesso em 3 de janeiro de 2012.

[22]http://httpd.apache.org/, Apache, **Apache**, acesso em 3 de janeiro de 2012.

[23]http://news.netcraft.com/archives/2011/05/02/may-2011-web-server-survey.html, *Netcraft,* ***Netcraft May 2011 Web Server Survey***, acesso em 3 de janeiro de 2012.

[24]http://www.mysql.com/, MySQL, **MySQL**, acesso em 3 de janeiro de 2012.

[25]www.mysql.com/products/enterprise/mysql-datasheet.en.pdfSimilar, MySQL, ***MySQL datasheet***, acesso em 3 de janeiro de 2012.

[26]http://www.wampserver.com/en/, WAMPSERVER, **WampServer**, acesso em 3 de janeiro de 2012.

[27]http://www.easyphp.org/, EasyPHP, **EasyPHP**, acesso em 3 de janeiro de 2012.

[28]http://tools.ietf.org/html/rfc2616, IETF, ***Request for Comments: 2616***, acesso em 1º de janeiro de 2012.

[29]http://www.w3.org/Protocols/rfc959/, *w3.org,* ***File Transfer Protocol (FTP)***, acesso em 3 de janeiro de 2012.

[30]http://www.w3.org/html/, World Wide Web Consortium (W3C), **News and Opinions About HTML**, acesso em 2 de janeiro de 2012.

[31]http://www.w3schools.com/.

[32]http://www.blumentals.net/htmlpad/, HTML Pad, ***HTML Pad***, acesso em 3 de janeiro de 2012.

[33] http://us3.php.net/manual/en/index.php, PHP.net, ***PHP Manual***, acesso em 02 de dezembro de 2011.

[33]http://www.php.net/, http://www.php.net/usage.php, PHP.net, ***Hypertext Preprocessor****,* acesso em 2 de janeiro de 2012.

[34]http://www.computerworld.com/s/article/43487/Application_Programming_Interface, Computer World, ***QuickStudy: Application Programming Interface (API)***, acesso em 3 de janeiro de 2012.

[35]http://code.google.com, Google Code, ***Google Code***, acesso em 18 de dezembro de 2011.

[36]http://code.google.com/apis/maps/, *Google Code,* ***Maps API Family***, acesso em 2 de janeiro de 2012.

[37]http://www.autoz.com.br/Shopping/Loja/Detalhes/default.asp?ProdutoID=19012&ItemID=1&Depto=23, Shopping AutoZ, **Rastreador e Bloqueador Vias 900-GPS da Svias**, acesso em 13 de janeiro de 2012.

[38]http://www.cellblock.com.br/track-moto.html, CellBlock Rastreadores e Bloqueadores, **Rastreador VTR200**, acesso em 13 de janeiro de 2012.

[39]http://www.lcjbrasil.com.br/loja/detalhes.asp?id=34&produto=104, Loja LCJ Brasil, KX500 Rastreador e Bloqueador Automotivo, acesso em 13 de janeiro de 2012.

[40]http://www.furuta.com.br/produto-49-rastreador_e_bloqueador_locktronic_rlk_sem_mensalidade, Furuta Car Audio, **Rastreador e bloqueador Locktronic RLK sem Mensalidade**, acesso dem 13 de janeiro de 2012.

[41]http://www.xsat.com.br/comprar/compraraspx/comprar.aspx, XSAT **Rastreador sem Mensalidade, Rastreador e Bloqueador PRO III**, acesso em 13 de janeiro de 2012.

[42]http://www.mpxshop.com/Produtos.asp?ProdutoID=1766&gclid=COG-kcfPz60CFUqb7Qod7Ticmg, MPX Shop, **Rastreador GPS GT01**, acesso em 13 de janeiro de 2012.

[43]http://www.dell.com/br/empresa/p/poweredge-tower-servers, Dell Brasil, **Servidores em Torre DELL PowerEdge**, acesso em 3 de janeiro de 2012.

http://us3.php.net/manual/en/index.php, PHP.net, ***PHP Manual***, acesso em 3 de janeiro de 2012.

http://www.w3schools.com/, W3Schools, ***Learn to create Websites***, acesso em 3 de janeiro de 2012.

8. ANEXOS

8.1. ANEXO A – Lista de comandos AT utilizados no projeto.

a. at+cfun=[0,1]
    i. 0: coloca o módulo em modo de carregar bateria
    ii. 1: coloca o módulo em modo de funcionamento total
b. at+creg? Verifica se há disponibilidade de rede
c. at+cpin? Verifica a disponibilidade do cartão SIM
d. at+csq Verifica a qualidade do sinal
e. at+cstt="tim.br","tim","tim" Escolhe a APN para conexão
f. at+cdnsorip=0 Decide se vai conectar em modo dns ou ip
g. at+ciicr Prepara conexão GPRS
h. at+cifsr Pergunta IP local
i. at+cipstart="tcp","200.17.220.150","80" Realiza uma conexão TCP/IP com o servidor, na porta desejada.
j. at+cipclose Fecha a conexão
k. at+cipstatus Verifica a estado da conexão
l. at+cpowd=1 Desliga o módulo
m. at+cops? Identifica a operadora de telefonia
n. +++ Quando conectado em modo transparente, entra em modo comando
o. ato Volta ao modo transparente
p. at+cmgf=1 Prepara codificação de mensagem
q. at+cmgs Envia mensagem sms
r. at+ciurc Omite a sentença "Call Ready"
s. at+gsn Pede o imei do módulo

8.2. ANEXO B – Guia do usuário KIT SIM900

SIM900_EVB kit_User Guide_V1.03.pdf

8.3. ANEXO C – *Datasheet* Módulo GPS 1000ME-RW

GPS_Module_1000ME-RW.pdf

8.4. ANEXO D – *Datasheet* microprocessador MPC89E515A

mpc89e515a.pdf

8.5. ANEXO E – Grupo de comandos AT para SIM300

SIM300C_ATC_V2.00.pdf

8.6. ANEXO F – *Hardware Design* SIM 300

SIM300C_HD_V2.04.pdf

8.7. ANEXO G – *Código do Website*

INDEX.PHP

```
<html>
<head>
  <title>Alarme Automotivo Inteligente</title>
  <link rel="stylesheet" type="text/css" href="index.css" />
```

```
</head>


<!--<?php
include "top.htm";
include "main.php";
?>-->



<frameset rows="30%,*">
  <frame scrolling=no frameborder=0 src="top.htm" />
  <frame name="showframe" scrolling=yes marginheight=0 frameborder=0
src="main.php" />
</frameset>


</html>
```

INDEX.CSS

```
*{
  margin: 0 auto;
}

a{
  text-decoration: none;
}

a:visited{
  color:#6600CC;
}

a:hover{
  color:#FFCC00;
}

#acesse{
  margin-top:10px;
  font-size:12px;
  font-weight:bold;
  font-family:verdana;
  color:black;
}
```

```
 body{
  background-color:#CCCCFF;
}

input{
  border:0;
  border-radius:5px;
}

#item{
  color:#6600CC;
  font-size:15px;
  font-family:verdana;
  color:black;
  list-style-type:none;
  text-align:left;
}

#login{
  width:400px;
  padding-left:5px;
  padding-top:10px;
  padding-bottom:10px;
  margin-top:5px;
```

```
  font-family: verdana;
}

#main{
  text-align:center;
  width:850px;
  height:100%;
  margin-top:20px;
  background-color:#5e71ea;
  border:dashed;
  border-color:#6633CC;
  border-width:1px;
  border-radius:10px;
}

#menu{
  text-align:center;
  width:915px;
  height:30px;

}

#postagem{
  text-align:justify;
  width:800px;
```

```
  height:100%;
  padding-left:10px;
  padding-top:10px;
  padding-bottom:10px;
  color:black;
  font-family:verdana;
}


#titulo{
 text-align:center;
}

#website{
  text-align:center;
  margin-top:10px;
  width:915px;
  height:100%;
}
```

MAIN.PHP

```
<html>
<head>
  <title>Alarme Automotivo Inteligente</title>
  <link rel="stylesheet" type="text/css" href="index.css" />
</head>

<body>
     <?php
      session_start();

      if ($_SESSION['logado'] == 1){

        echo
        @include('logado_index.php');}

        else{
        echo"<DIV id=main><DIV id=postagem>

<br />Bem vindo ao portal AAI.
    <DIV id=login>
    <form action='recebe_login.php' method=POST>
    <table>
```

```
    <tr><td>Usuario</td><TD><input type=text name='login'
maxlenght='10'/></TD></TR>

    <tr><td>Senha</td><TD><input type=password name='senha_log' /> </TD></TR>

    <tr><TD></TD><td><input type='submit' value='Login'/></TD></TR>

    </table>

    </form>

    </DIV>


    Primeiro acesso? <a href='cadastro_usuario.php'>Cadastre-se.</a>

    </DIV></div>

   ";}

   ?>


</body>

</html>
```

CADASTRO_USUARIO.PHP

```
<html>
<head>
  <title>Alarme Automotivo Inteligente</title>
  <link rel="stylesheet" type="text/css" href="index.css" />
</head>

    <td><DIV ID=main>

        <DIV id=postagem>

        Cadastro:<br />

        <form action='recebe_cadastro.php' method=POST>

        <table>
        <tr><td>Nome completo:</td><TD><input type=text name='nome' maxlenght='40'/></TD></TR>
        <tr><td>Nome de acesso (usuario)<sup>1</sup>:</td><TD><input type=text name='usuario' maxlenght='10'/></TD></TR>
        <tr><td>Senha:</td><TD><input type=password name='senha' maxlenght='20'/> </TD></TR>
        <tr><td>Digite sua senha novamente:</td><TD><input type=password name='senha2' maxlenght='10'/> </TD></TR>
        <tr><td>CPF<sup>2</sup>:</td><TD><input type=text name='cpf' maxlenght='12'/> </TD></TR>
```

```
<tr><td>Sexo:</td><TD><input type="radio" name="sexo" value="homem" />
Maculino<br /></TD></TR>

<tr><TD></TD><td><input type="radio" name="sexo" value="mulher" />
Feminino</TD></TR>

<tr><td>Telefone (DDD + Numero)</td><TD><input type=text name='telefone'
maxlenght='10'/> </TD></TR>

<tr><TD></TD><td><input type="submit" value="Cadastre!"/></TD></TR>

</table></form>

<br />

<font size=2px>[1]: Seu nome de acesso deve conter no maximo 10 caracteres.<br
/></font>

<font size=2px>[2]: Digite só numeros, sem "-" ou ".".<br /></font>

</DIV>

</DIV></td>



</html>
```

RECEBE_CADASTRO.PHP

```
<?php
 $con = mysql_connect("localhost","erica","72hrGHYE6r2ArBR5");
 if (!$con) { die('Não foi possivel estabelecer conexao: ' . mysql_error());  }
 $db_selected = mysql_select_db('erica', $con);
if (!$db_selected) {
    die ('Não foi possivel estabelecer conexao: ' . mysql_error());}

 $nome = ($_POST['nome']);
 $usuario = ($_POST['usuario']);
 $senha = ($_POST['senha']);
 $senha2 = ($_POST['senha2']);
 $cpf = ($_POST['cpf']);
 $sexo = $_POST['sexo'];
 $telefone = $_POST['telefone'];

   #//verificar se o usuario ja existe
   $s=mysql_query("SELECT * FROM usuarios WHERE usuario = '$usuario'");
   $busca=mysql_numrows($s);
   if($busca!=0){ echo @include("insucesso1_cadastro.php");
   $erro++;  }

   #//verificar se senhas são iguais
   if($senha != $senha2){
   echo @include("insucesso2_cadastro.php");
```

```
$erro++;}

#//encontrar caracteres no cpf
if( ereg("[()-.,:;*&¬!?|+}{/]", $cpf) ){
echo @include("insucesso3_cadastro.php");
$erro++; }

#//verificar se o usuario nao tem mais de 10 caracteres
if(strlen($usuario)>10)             {
echo @include("insucesso3_cadastro.php");
$erro++; }

#//verificar se campos não são nulos
if(empty($nome)){
echo @include("insucesso3_cadastro.php");
$erro++; }

if(empty($usuario)){
echo @include("insucesso3_cadastro.php");
$erro++; }

if(empty($senha)){
echo @include("insucesso3_cadastro.php");
$erro++; }
```

```
   if(empty($senha2)){
   echo @include("insucesso3_cadastro.php");
   $erro++; }

   if(empty($cpf)){
   echo @include("insucesso3_cadastro.php");
   $erro++; }

   if(empty($sexo)){
   echo @include("insucesso3_cadastro.php");
   $erro++; }

   if(empty($telefone)){
   echo @include("insucesso3_cadastro.php");
   $erro++; }
if($erro==0){
#//inserir no banco de dados se tudo for OK
mysql_query("INSERT INTO usuarios (nome, usuario, senha, cpf, sexo, telefone)
VALUES ('$nome','$usuario','$senha','$cpf','$sexo', '$telefone')");
    echo '<script type="text/javascript">alert("Cadastro realizado com
sucesso.")</script>';
    @include ("main.php");}
mysql_close($con)
?>
```

RECEBE_LOGIN.PHP

```
<?
 $con = mysql_connect("localhost","erica","72hrGHYE6r2ArBR5");
 if (!$con) { die('Não foi possivel estabelecer conexao: ' . mysql_error());  }

 $db_selected = mysql_select_db('erica', $con);
 if (!$db_selected) {
    die ('Não foi possivel estabelecer conexao: ' . mysql_error()); }

$login = ($_POST['login']);
$db_user = mysql_query("SELECT * FROM usuarios WHERE usuario='$login'");
$senha_log = ($_POST['senha_log']);

    if(mysql_num_rows($db_user) == 1) {

        $query = mysql_query("SELECT * FROM usuarios WHERE usuario='$login'");
        $dados = mysql_fetch_array($query);
        if($senha_log == $dados['senha']) {
            $_SESSION['nome'] = $dados['nome'];
            $_SESSION['id'] = $dados['id'];
            $_SESSION['sexo'] = $dados['sexo'];
            session_start();
             $_SESSION["logado"]=1;
            echo
```

```
            @include ("logado_index.php");
            exit;
        } else {
            echo '<script type="text/javascript">alert("Senha incorreta.")</script>';
            @include ('main.php');
            exit;
        }
    } else {
            echo '<script type="text/javascript">alert("Usuário invalido.")</script>';
            @include ('main.php');
             exit;
    }
?>
```

LOGADO_INDEX.PHP

```
<html>
 <head>
  <title>Alarme Automotivo Inteligente</title>
  <link rel="stylesheet" type="text/css" href="index.css" />
</head>
<body><DIV ID=main>
  <?php
   $con = mysql_connect("localhost","erica","72hrGHYE6r2ArBR5");
 if (!$con) { die('Não foi possivel estabelecer conexao: ' . mysql_error());  }

 $db_selected = mysql_select_db('erica', $con);
 if (!$db_selected) {
    die ('Não foi possivel estabelecer conexao: ' . mysql_error()); }
    session_start();
    $query = mysql_query("SELECT * FROM veiculos WHERE id='.S_SESSION['id']."');

    if ($_SESSION['logado'] == 1){
        if($_SESSION['sexo']== mulher){
        echo "
        <DIV id=postagem>
        Bem vinda <b>".$_SESSION['nome']."</b> (ID #".$_SESSION['id'].").";}
        else {
        echo "<DIV id=postagem>
        Bem vindo <b>".$_SESSION['nome']."</b> (ID #".$_SESSION['id']).";  }
```

```
    echo
    "<br /> <br />    Por favor selecione o que deseja fazer:
     <br /><br />
    <form action='jump.php' method='post'>
    <select name=url>
    <option value='cadastro_veiculo.php'>Cadastrar novo veiculo</option>
    <option value='listar_veiculo.php'>Listar veiculos cadastrados</option>
    <option value='alterar_usuario.php'>Alterar informacoes de usuario</option>
    </select>
    <input type='submit' value='Ir'/>
    </form>
    <br /><br /><br /><br /><form action='logoff.php'>
    <input type='submit' value='Encerrar sessao'>
    </form>
     </DIV>
     ";
   }else{
   echo "<DIV id=postagem>
    É preciso estar logado para entrar nesta pagina.</div>";
   }
 ?>
 </DIV>
 </body>
</html>
```

CADASTRO_VEICULO.PHP

```
<html>
<head>
  <title>Alarme Automotivo Inteligente</title>
  <link rel="stylesheet" type="text/css" href="index.css" />
</head>

     <?php
   $con = mysql_connect("localhost","erica","72hrGHYE6r2ArBR5");
 if (!$con) { die('Não foi possivel estabelecer conexao: ' . mysql_error());  }

 $db_selected = mysql_select_db('erica', $con);
 if (!$db_selected) {
    die ('Não foi possivel estabelecer conexao: ' . mysql_error()); }

    session_start();

      if ($_SESSION['logado'] == 1){
        echo "    <td><DIV ID=main>

        <DIV id=postagem>
        Cadastro de novos veiculos:<br />
        <form action='recebe_veiculo.php' method=POST>
```

```
        <table>
        <tr><td>Modulo:</td><TD><input type=text name='modulo'
maxlenght='40'/></TD></TR>
        <tr><td>Placa:</td><TD><input type=text name='placa'
maxlenght='40'/></TD></TR>
        <tr><td>Modelo:</td><TD><input type=text name='modelo' maxlenght='20'/>
</TD></TR>
        <tr><td>Cor:</td><TD><input type=text name='cor' maxlenght='10'/> </TD></TR>
        <tr><TD></TD><td><input type='submit' value='Cadastre!'/></TD></TR>
        </table></form>

        <form action='logado_index.php'>
        <input type='submit' value='Retornar'>
        </form>

    </DIV>
    </DIV></td>
        '; }else{
       echo "<DIV id=postagem>
        É preciso estar logado para entrar nesta pagina.</div>";
       }

    ?>
</html>
```

LISTAR_VEICULO.PHP

```
<html>
 <head>
  <title>Alarme Automotivo Inteligente</title>
  <link rel="stylesheet" type="text/css" href="index.css" />
</head>
<body><DIV ID=main>


     <?php
   $con = mysql_connect("localhost","erica","72hrGHYE6r2ArBR5");
 if (!$con) { die('Não foi possivel estabelecer conexao: ' . mysql_error());  }


 $db_selected = mysql_select_db('erica', $con);
 if (!$db_selected) {
    die ('Não foi possivel estabelecer conexao: ' . mysql_error()); }
    session_start();


    $id = ($_SESSION['id']);
    $query = "SELECT * FROM veiculos WHERE id='$id'";
    $result = mysql_query($query);
    $num = mysql_numrows($result);


     if ($_SESSION['logado'] == 1){
       if($num == 0){
          echo "<div id=postagem>
```

```
          Não há veiculos cadastrados para ".$_SESSION['nome'].".
         <br /><br /><form action='logado_index.php'>
        <input type='submit' value='Retornar'>
        </form></div>";
          }else {
            echo "<div id=postagem>";
            while($row= mysql_fetch_assoc($result))
                {  $modulo=$row['modulo'];
        echo "
         <b>Modulo</b> : {$row['modulo']} <br />
         <b>Placa</b> : {$row['placa']} <br />
         <b>Modelo</b> : {$row['modelo']} <br />
         <b>Cor</b> : {$row['cor']} <br />
        <a href='editar_veiculo.php?modulo=$modulo'><img src='b_edit.png' width='20'
height='20' title='Editar' /></a>
        <a href='deletar_veiculo.php?modulo=$modulo'><img src='b_delete.png'
width='20' height='20' title='Excluir' /></a>
        <a href='rastrear_veiculo2.php?modulo=$modulo'><img src='b_rastrear.png'
width='20' height='20' title='Rastrear posicao' /></a><br /><br />"    }
          echo "<br /><br /><form action='logado_index.php'>
        <input type='submit' value='Retornar'>
        </form></div>";}     }
    ?>
   </DIV>
    </body>
   </html>
```

RASTREAR_VEICULO.PHP

```
<html>
<head>
  <title>Alarme Automotivo Inteligente</title>
  <link rel="stylesheet" type="text/css" href="index.css" />
</head>

    <?php
    $con = mysql_connect("localhost","erica","72hrGHYE6r2ArBR5");
    if (!$con) { die('Não foi possivel estabelecer conexao: ' . mysql_error());  }

    $db_selected = mysql_select_db('erica', $con);
    if (!$db_selected) {
    die ('Não foi possivel estabelecer conexao: ' . mysql_error()); }

    session_start();

      if ($_SESSION['logado'] == 1){

      $modulo = $_GET['modulo'];
      $_SESSION['modulo'] = $modulo;
      $query  = "SELECT * FROM veiculos WHERE modulo='$modulo'";
      $result = mysql_query($query);
      $row= mysql_fetch_assoc($result);
```

```
    $query2 = "SELECT * FROM posicao WHERE modulo='$modulo'";
    $result2 = mysql_query($query2);
    $num = mysql_numrows($result2);

    $query3 = "SELECT * FROM posicao WHERE modulo='$modulo' ORDER BY
datahora DESC LIMIT 1";
    $result3 = mysql_query($query3);

          if($num == 0){
          echo "<div id=main><div id=postagem>
          Não há posicoes cadastradas para o modulo ".$_SESSION['modulo'].".
          <br /><br /><form action='listar_veiculo.php'>
          <input type='submit' value='Retornar'>
          </form></div></div>";

         }else {

           $row3= mysql_fetch_assoc($result3);
           $lat = $row3['latitude'];
           $long = $row3['longitude'];

           echo "
    <meta name='viewport' content='initial-scale=1.0, user-scalable=no' />
    <script type='text/javascript'
src='http://maps.google.com/maps/api/js?sensor=true'></script>
```

```
<script type='text/javascript>
function maps() {
var latlng = new google.maps.LatLng($lat, $long);
var myOptions = {
  zoom: 13,
  center: latlng,
  mapTypeId: google.maps.MapTypeId.ROADMAP   };

      var map = new google.maps.Map(document.getElementById('map_canvas'),
myOptions);";

      $f = 0;
      $d = 0;
      while(
      $row2= mysql_fetch_assoc($result2)){
      $latitude = $row2['latitude'];
      $longitude = $row2['longitude'];
      $f++;
      $d++;

      echo "
var image = 'posi$d.png';
var myLatLng = new google.maps.LatLng($latitude, $longitude);
var beachMarker$f = new google.maps.Marker({
position: myLatLng,
```

```
map: map,
icon: image,         })  ";}

echo "
}</script>
<body onload='maps()'><div id=main><div id=postagem>
Veiculo Rastreado:<br /><br />

   <b>Modulo</b> :{$row['modulo']} <br>
   <b>Placa</b> : {$row['placa']} <br>
   <b>Modelo</b> : {$row['modelo']} <br>
   <b>Cor</b> : {$row['cor']}
   <br /><br />
<div id='map_canvas' style='width:60%; height:70%'></div>
<form action='listar_veiculo.php'>
<input type='submit' value='Retornar'>
</form></div></div></body>";  }
}else{
echo "<DIV id=postagem>
É preciso estar logado para entrar nesta pagina.</div>";    }
?>

</html>
```

LISTAR_POSIÇÃO.PHP

```
<html>
<head>
  <title>Alarme Automotivo Inteligente</title>
  <link rel="stylesheet" type="text/css" href="index.css" />
</head>
    <?php
    $con = mysql_connect("localhost","erica","72hrGHYE6r2ArBR5");
    if (!$con) { die('Não foi possivel estabelecer conexao: ' . mysql_error());  }

    $db_selected = mysql_select_db('erica', $con);
    if (!$db_selected) {
    die ('Não foi possivel estabelecer conexao: ' . mysql_error()); }

    session_start();
      if ($_SESSION['logado'] == 1){

      $modulo = $_GET['modulo'];
      $_SESSION['modulo'] = $modulo;
      $query  = "SELECT * FROM veiculos WHERE modulo='$modulo'";
      $result = mysql_query($query);
      $row= mysql_fetch_assoc($result);
```

```
$query2 = "SELECT * FROM posicao WHERE modulo='$modulo' ORDER BY datahora DESC";

$result2 = mysql_query($query2);

$num = mysql_numrows($result2);

$query3 = "SELECT * FROM posicao WHERE modulo='$modulo' ORDER BY datahora DESC LIMIT 1";

$result3 = mysql_query($query3);


    if($num == 0){

    echo "<div id=main><div id=postagem>

    Não há posicoes cadastradas para o modulo ".$_SESSION['modulo'].".

    <br /><br /><form action='listar_veiculo.php'>

    <input type='submit' value='Retornar'>

    </form></div></div>";

   }else {

     $row3= mysql_fetch_assoc($result3);

     $lat = $row3['latitude'];

     $long = $row3['longitude'];

     echo "

<meta name='viewport' content='initial-scale=1.0, user-scalable=no' />

<script type='text/javascript' src='http://maps.google.com/maps/api/js?sensor=true'></script>

<script type='text/javascript'>

function maps() {

var latlng = new google.maps.LatLng($lat, $long);

var myOptions = {
```

```
  zoom: 14,
  center: latlng,
  mapTypeId: google.maps.MapTypeId.ROADMAP    };'';
        $f = ($num);
        while(
        $row2= mysql_fetch_assoc($result2) AND $f !=0){
        $latitude = $row2['latitude'];
        $longitude = $row2['longitude'];
        $datahora = $row2['datahora'];
        $f--;
        echo ''
    var myLatLng$f = new google.maps.LatLng($latitude, $longitude);
       var infowindow$f = new google.maps.InfoWindow();
       var geocoder$f;
       geocoder$f = new google.maps.Geocoder();
       geocoder$f.geocode({'latLng':myLatLng$f}, function(results, status) {
      if (status == google.maps.GeocoderStatus.OK) {
          if (results[1]) {
          document.getElementById('posi$f').innerHTML=(results[1].formatted_address);
          document.getElementById('hora$f').innerHTML='$datahora';
            } else { alert('No results found'); }
      } else { alert('Geocoder failed due to: ' + status);  }  });
';}
echo ''
}</script>
```

```
    <body onload='maps()'><div id=main><div id=postagem>
    Veiculo Rastreado:<br /><br />
        <b>Modulo</b> :{$row['modulo']} <br>
        <b>Placa</b> : {$row['placa']} <br>
        <b>Modelo</b> : {$row['modelo']} <br>
        <b>Cor</b> : {$row['cor']}
        <br /><br />
         <table border=1px bgcolor=#CCCCFF>
        <tr><td><b>Ponto</b></td><td><b>Endereço</b></td><td><b>Data e
hora</b></td></tr>";
        $d = ($num-1);
        while ($d !=-1) {
        echo "<tr><td>Ponto$d</td><td><p id='posi$d'></p></td><td><p
id='hora$d'></p></td></tr>";
        $d--;}

        echo "</table><br />
    <div id='map_canvas'></div>
    <form action='listar_veiculo.php'>
    <input type='submit' value='Retornar'>
    </form></div></div></body>";    }
    }else{
    echo "<DIV id=postagem>
    É preciso estar logado para entrar nesta pagina.</div>";}
    ?>
</html>
```

LOGOFF.PHP

```
<?php
unset($_SESSION['logado']);

@include("main.php");
?>
```

8.8. ANEXO H – *Código Gravado no Microcontrolador*

```
; Linguagem: Assembly

;
;       ****************************************************************
;       *                          MAIN.S03                            *
;       *       Controle de conexão do módulo GPRS SIM340CZ            *
;       *       Elaborado por Helder Avelar, a partir de 12/09/11      *
;       *                                                              *
;       ****************************************************************

;-----------------------------------------------------------------------;
;               Area reservada para vetores de interrupcao              ;
;-----------------------------------------------------------------------;

        org     addr_zero+03h           ;int0
        lcall   int_rx1
        org     addr_zero+07h
        dw      0
        org     addr_zero+09h
        org     addr_zero+0bh
        ljmp    int_timer0          ;int
```

```
;       reti
        org     addr_zero+13h           ; int1
        reti                            ;edit
        org     addr_zero+1bh
        reti                            ;edit
        org     addr_zero+23h
        ljmp    int_serial              ;int RI/TI
        org     addr_zero+2bh           ;int overflow timer 2
        ljmp    int_timer2
        org     addr_zero+0060h         ;Inicio do Codigo


inic_relig:

;-------------------------------------------------------------------------;
; Inicializacoes dos Latchs                                               ;
;-------------------------------------------------------------------------;

      mov     psw,#00h        ; Seleciona sempre banco 0
        mov     P0, #00h
        mov     P1, #00h
        mov     P2, #0FFh
        mov     P3, #0FFh
        mov     v_sistema1, #00
        mov     v_sistema2, #00


;-------------------------------------------------------------------------;
; Inicializa Temporizadores e Interrupcoes (Menos a Serial)               ;
;-------------------------------------------------------------------------;
```

```
        lcall   inic_temp

nao_zerar:
        ret


;-------------------------------------------------------------------;
;                   Rotina de iniciar quando disparar alarme                    ;
;-------------------------------------------------------------------;

rot_000:
        setb    p_led2
        setb    p_led1
        setb    p_led0
        mov     P0,#00
        clr     b_chave
espera_pulso:
        jnb     p_start,espera_pulso
        lcall   tempo500ms
        lcall   tempo500ms
        lcall   tempo500ms
        lcall   tempo500ms
        jb      p_start,rot_001
        jb      p_start,rot_001
        lcall   tempo100ms
        jb      p_start,rot_001
        jb      p_start,rot_001
        jmp     rot_000

;-------------------------------------------------------------------;
;                   Rotina de inicialização do modem                    ;
```

```
;-------------------------------------------------------------------;

rot_001:
        mov         sp,#v_fim           ; Stack de memória
        mov         v_count, #00
        mov         v_mseg, #00
        mov         v_mseg2, #00
        mov         v_seg, #00
        clr         c
        setb        b_emajuste

        lcall   inic_relig

        mov   p1,#00
        mov   P2,#0FFh
        mov   P3,#0FFh
        lcall   tempo500ms
        clr     p_led0
        lcall   tempo500ms
        clr     p_led1
        lcall   tempo500ms
        clr     p_led2
        lcall   tempo500ms
        mov   P3, #0FFh
        mov   v_reset,#00
        jb      b_chave,rot_002
        lcall   conecta_gprs
        mov   v_count,#01
        ljmp    rotina

rot_002:
```

```
        lcall   tempo100ms
        lcall   cpowd1
        lcall   tempo500ms
        ljmp    rot_000

;-------------------------------------------------------------------;
;       Rotina - Executa aqui sempre, em funcionamento como modem gprs ;
;-------------------------------------------------------------------;
rotina:
        jb      b_chave,rot_002
        clr     b_conectando
        mov     P1,#00
        clr     p_led0
        setb    p_led2
        lcall   tempo500ms
        clr     p_led2
        setb    p_led1
        lcall   tempo500ms
        setb    p_led0
        clr     p_led1
        lcall   tempo500ms
        mov     v_loop,#00

rot_050:
        setb    EX0
        lcall   tempo500ms
        lcall   tempo500ms
        lcall   tempo500ms
        lcall   tempo500ms
        clr     EX0
```

```
        lcall   valida_gps

        jnb     b_gpsok,rot_050

        mov     dptr,#z_bufgps+23
        movx    a,@dptr
        mov     v_gpsh,a
        mov     dptr,#z_bufgps+36
        movx    a,@dptr
        mov     v_gpsv,a


rot_070:
        lcall   dois_min

        lcall   limpa_buffer
        jb      b_chave,rotina
        lcall   cipstart
        lcall   wait_response
        setb    EX0

        lcall   tempo500ms
        lcall   tempo500ms
        lcall   tempo500ms
        lcall   tempo500ms

        clr     EX0

        lcall   valida_gps
        jnb     b_gpsok,rot_070
```

```
        jb      b_msgenviada,rot_070
        mov     dptr,#z_bufgps+23
        movx    a,@dptr
        cjne    a,v_gpsh,rot_071
        mov     dptr,#z_bufgps+36
        movx    a,@dptr
        cjne    a,v_gpsh,rot_071
        jmp     rot_070

rot_071:
        mov     v_mensagem,#02
        lcall   envia_sms
        lcall   limpa_buffer
        setb    b_msgenviada
        jmp     rot_070

dois_min:
        push    b
        mov     b,#4
dois_min100:
        lcall   trinta_sec
        djnz    b,dois_min100
        pop     b
        ret

trinta_sec:
        push    b
        mov     b,#15
trinta_sec100:
        clr     p_led0
        setb    p_led2
```

```
        lcall   tempo500ms
        clr     p_led2
        setb    p_led1
        lcall   tempo500ms
        setb    p_led0
        clr     p_led1
        lcall   tempo500ms
        djnz    b,trinta_sec100
        pop     b
        ret

        mov     v_count,#4              ; pra não entrar nunca na rotina
        djnz    v_count, rot090
        setb    EX0
        lcall   tempo500ms
        lcall   tempo500ms
        clr     EX0

        push    dpl
        push    dph
        push    b
        mov     dptr,#z_bufgps
        mov     b,#20
loopgps:
        lcall   tx_serial
        inc     dptr
        djnz    b,loopgps
        pop     b
        pop     dph
        pop     dpl
        mov     v_count,#5
```

```
rot090:
        mov     a,v_reset
        cjne    a,#03,rot100
        mov     v_reset,#00
        setb    b_restart_ip
rot100:
        jnb     b_ver_conect, rot200
        lcall   tempo100ms
        setb    b_verificando
        mov     a,#02Bh
        lcall   tx_serial
        lcall   tx_serial
        lcall   tx_serial
        lcall   tempo500ms
        lcall   tempo500ms
        lcall   limpa_buffer
        setb    b_conectando
        lcall   cipstatus
        mov     v_timeout,#10
        lcall   wait_response
        clr     b_conectando
        jnb     b_timeout,rot111
        setb    b_restart_modem
        clr     b_timeout
rot111:
        lcall   tempo500ms
        mov     a,#'a'
        lcall   tx_serial
```

```
        mov   a,#'t'
        lcall tx_serial
        mov   a,#'o'
        lcall tx_serial
        lcall fim_comando
        lcall tempo100ms
        clr   b_verificando
        lcall ver_resposta
        lcall tempo500ms
        lcall tempo500ms
rot200:
        jnb   b_ser_ok,rot300
        clr   b_ser_ok
        mov   R0,#z_string
        mov   a,#'P'
        mov   @R0,a
        inc   R0
        mov   a,#'D'
        mov   @R0,a
        inc   R0
        mov   a,#'P'
        mov   @R0,a
        mov   a,#3
        mov   b,#100
        lcall procura_string
        jnz   rot250
        setb  b_restart_modem
rot250:
        mov   R0,#z_string
        mov   a,#'C'
        mov   @R0,a
```

```
        inc     R0
        mov     a,#'L'
        mov     @R0,a
        inc     R0
        mov     a,#'O'
        mov     @R0,a
        inc     R0
        mov     a,#'S'
        mov     @R0,a
        inc     R0
        mov     a,#'E'
        mov     @R0,a
        mov     a,#5
        mov     b, #20h
        lcall   procura_string
        lcall   limpa_buffer        ; EDIT
        jnz     rot300
        setb    b_restart_ip
rot300:
        jnb     b_restart_ip,rot400
        lcall   tempo100ms
        mov     a,#02Bh
        lcall   tx_serial
        lcall   tx_serial
        lcall   tx_serial
        lcall   tempo500ms
        lcall   tempo500ms
        lcall   cipclose            ; EDIT
        lcall   tempo500ms
        lcall   tempo500ms
        lcall   conect_370
```

```
rot400:
        jnb     b_restart_modem,rot500
        lcall   tempo100ms
        mov     a,#02Bh
        lcall   tx_serial
        lcall   tx_serial
        lcall   tx_serial
        lcall   tempo500ms
        lcall   tempo500ms
        lcall   cpowd1
        mov     v_loop,#30
rot410:
        lcall   tempo500ms
        djnz    v_loop,rot410
        clr     b_restart_ip
        clr     b_restart_modem
        lcall   conecta_gprs
rot500:
rot900:
        jmp     rotina

ver_resposta:                   ;verifica qual foi a resposta do
ipstatus e prepara tratamento
        mov     R0,#z_string
        mov     a,#'C'
        mov     @R0,a
        inc     R0
        mov     a,#'O'
        mov     @R0,a
        inc     R0
        mov     a,#'N'
```

```
        mov   @R0,a
        inc   R0
        mov   a,#'N'
        mov   @R0,a
        inc   R0
        mov   a,#'E'
        mov   @R0,a
        inc   R0
        mov   a,#'C'
        mov   @R0,a
        inc   R0
        mov   a,#'T'
        mov   @R0,a
        inc   R0
        mov   a,#' '
        mov   @R0,a
        inc   R0
        mov   a,#'O'
        mov   @R0,a
        inc   R0
        mov   a,#'K'
        mov   @R0,a
        mov   a,#10
        mov   b, #20h
        lcall procura_string
        jz    ver_resp900

ver_resp100:
        mov   R0,#z_string
        mov   a,#'C'
        mov   @R0,a
```

```
        inc     R0
        mov     a,#'L'
        mov     @R0,a
        inc     R0
        mov     a,#'O'
        mov     @R0,a
        inc     R0
        mov     a,#'S'
        mov     @R0,a
        inc     R0
        mov     a,#'E'
        mov     @R0,a
        mov     a,#5
        mov     b, #20h
        lcall   procura_string
        jz      ver_resp700

ver_resp200:
        mov     R0,#z_string
        mov     a,#'P'
        mov     @R0,a
        inc     R0
        mov     a,#'D'
        mov     @R0,a
        inc     R0
        mov     a,#'P'
        mov     @R0,a
        mov     a,#3
        mov     b, #20h
        lcall   procura_string
        jz      ver_resp800
```

```
ver_resp300:
        mov     R0,#z_string
        mov     a,#'I'
        mov     @R0,a
        inc     R0
        mov     a,#'N'
        mov     @R0,a
        inc     R0
        mov     a,#'I'
        mov     @R0,a
        inc     R0
        mov     a,#'T'
        mov     @R0,a
        inc     R0
        mov     a,#'I'
        mov     @R0,a
        inc     R0
        mov     a,#'A'
        mov     @R0,a
        inc     R0
        mov     a,#'L'
        mov     @R0,a
        mov     a,#7
        mov     b, #20h
        lcall   procura_string
        jz      ver_resp800

ver_resp700:
        setb    b_restart_ip
        jmp     ver_resp900
```

```
ver_resp800:
        setb    b_restart_modem
ver_resp900:
        mov     v_minuto, #5
        clr     b_ver_conect
        ret

;*****************************************************************************************
**********
;
;       Procura string em z_buffer
;       Entrada: acc (tamanho string), z_string (string a ser procurada), z_buffer
(string na qual será procurada)
;       Saída: acc = 0 se ok, <> 0 se erro
;*****************************************************************************************
**********
procura_string:
        push    v_calc_n1
        push    v_calc_n2
        mov     v_calc_n1, a            ; Salva quantidade de digitos
        mov     dptr, #z_buffer
proc_st100:
        mov     R0,#z_string
proc_st110:
        mov     a, @R0
        mov     v_calc_n2, a
        movx    a, @dptr
        xrl     a, v_calc_n2                    ; Compara com o primeiro digito
        jnz     proc_st190                      ; não é cai fora
proc_st120:
        push    v_calc_n1
```

```
        dec     v_calc_n1
proc_st125:
        inc     R0
        inc     dptr
        mov     a, @R0
        mov     v_calc_n2, a
        movx    a, @dptr
        xrl     a, v_calc_n2                    ; Compara com a primeiro digito
        jnz     proc_st180                      ; não é cai fora
        dec     b
        mov     a, v_calc_n1
        djnz    v_calc_n1, proc_st125
proc_st130:
        pop     v_calc_n1
        mov     a, #00
        jmp     proc_st900
proc_st180:
        pop     v_calc_n1
        jmp     proc_st192
proc_st190:
        inc     dptr
proc_st192:
        djnz    b, proc_st100
        mov     a, #01                  ;não achou
proc_st900:
        pop     v_calc_n2
        pop     v_calc_n1
        ret

;*****************************************************************************************
**********
```

```
;
;       Procura string em z_buffer
;       Entrada: acc (tamanho string), z_string (string a ser procurada = GPGGA),
z_bufgps (string na qual será procurada)
;       Saída: acc = 0 se ok, <> 0 se erro
;*****************************************************************************************
**********
procura_gps:
        push   v_calc_n1
        push   v_calc_n2
        mov    v_calc_n1, a          ; Salva quantidade de digitos
        mov    dptr, #z_bufgps
proc_gps100:
        mov    R0,#z_string
proc_gps110:
        mov    a, @R0
        mov    v_calc_n2, a
        movx   a, @dptr
        xrl    a, v_calc_n2                    ; Compara com o primeiro digito
        jnz    proc_gps190                     ; não é cai fora
proc_gps120:
        push   v_calc_n1
        dec    v_calc_n1
proc_gps125:
        inc    R0
        inc    dptr
        mov    a, @R0
        mov    v_calc_n2, a
        movx   a, @dptr
        xrl    a, v_calc_n2                    ; Compara com a primeiro digito
        jnz    proc_gps180                     ; não é cai fora
```

```
        dec     b
        mov     a, v_calc_n1
        djnz    v_calc_n1, proc_gps125
proc_gps130:
        pop     v_calc_n1
        mov     a, #00
        jmp     proc_gps900
proc_gps180:
        pop     v_calc_n1
        jmp     proc_gps192
proc_gps190:
        inc     dptr
proc_gps192:
        djnz    b, proc_gps100
        mov     a, #01                  ;não achou
proc_gps900:
        pop     v_calc_n2
        pop     v_calc_n1
        ret

;-------------------------------------------------------------------------;
;       Tratamento da interrupcao automatica                              ;
;       Recarga com 0:                                                    ;
;-------------------------------------------------------------------------;

int_timer0:
        push            IE
        jnb             p_chave,it0_900
        setb            b_chave
it0_900:
        pop             IE
```

```
        reti


;-------------------------------------------------------------------------;
;       Tratamento da interrupcao automatica (timer2)                     ;
;       Recarga com 0:                                                    ;
;         clock 22,1184 MHz / 6 / 65536 = 56 interrupcoes por segundo (138us)
        ;
;-------------------------------------------------------------------------;

int_timer2:
        push    psw
        push    acc
        push    v_reg0
        push    IE
        clr     TF2
        mov     TH2, #06Fh
        mov     TL2, #0FFh          ;overflow a cada 10ms

it2_100:
        mov     a, v_timeout
        jz      it2_900
        dec     aux_cont            ;aux_cont vai zerar a cada 1000ms
        mov     a,aux_cont
        jnz     it2_900
;it2_200:
        mov     aux_cont,#60h
        djnz    v_timeout,it2_900   ;v_timeout vai decrementar a cada 1s
        jb      b_conectando,it2_900
        mov     a, v_minuto
        jz      it2_900
```

```
        mov     v_timeout,#60
        djnz    v_minuto,it2_900
        setb    b_ver_conect
        inc     v_reset

it2_900:
        pop     IE
        pop     v_reg0
        pop     acc
        pop     psw
        reti



;*********************************************************************
;*                                                                   *
;*      INICIO DAS ROTINAS DE CONEXÃO E CONTROLE DO MÓDULO GPRS      *
;*                                                                   *
;*********************************************************************


;-------------------------------------------------------------------;
;               Rotina para conectar GPRS                           ;
;-------------------------------------------------------------------;

conecta_gprs:
        setb    b_conectando
        mov     P1,#00
        setb    p_led1
        lcall   tempo200ms
```

```
        clr     p_led1
        setb    p_led2
        lcall   tempo200ms
        clr     p_led2
        setb    p_led0
        lcall   tempo200ms
        clr     p_led0
        lcall   tempo200ms

        jmp   conect_130        ; considerando que o módulo sempre vai ligar, pula direto pro inicio das configs

conect_130:
        mov   v_timeout,#10
        clr   b_zera_cfun       ; coloca o módulo em modo de comunicação total
        lcall limpa_buffer
        lcall cfun1
        lcall wait_response
        lcall tempo100ms
        lcall ciurc
        lcall tempo500ms
conect131:
        mov   R0,#z_string
        mov   a,#'O'
        movx  @R0,a
        inc   R0
        mov   a,#'K'
        movx  @R0,a
        mov   a,#02
        mov   b,#40h
```

```
        lcall   procura_string
;       jz      conect_133
;       ljmp    conecta_gprs
conect_133:
        mov     b,#10
conect_135:
        lcall   tempo500ms
        djnz    b,conect_135

        mov     v_timeout,#10h
conect_136:
        lcall   limpa_buffer
        lcall   csq
        lcall   wait_response
        jnb     b_timeout,conect_137
        jmp     conect_133          ; tenta de novo
;       setb    b_restart_modem
;       ret
conect_137:
        lcall   tempo100ms
        mov     dptr,#z_buffer+8
        movx    a,@dptr
        cjne    a,#'9',conect_137a  ; se for '9', o valor ainda não foi descoberto.
        jmp     conect_136          ; tenta de novo
conect_137a:
        mov     dptr,#z_buffer+9
        movx    a,@dptr
        cjne    a,#',',conect_138   ; se for ',', o valor está abaixo de 10, portanto
inaceitavel.
        jmp     conect_136          ; tenta de novo
```

```
conect_138:
        mov     v_mensagem,#01

        jb      b_chave,conect_cancel

        lcall   envia_sms
        lcall   cdnscfg
        jmp     conect_139

conect_cancel:
        ljmp    rot_002

conect_139:
        mov     v_timeout,#020h
conect_140:
        lcall   tempo500ms
        lcall   limpa_buffer
        lcall   cops                    ; manda cops, para ver em qual operadora está
conectado
        lcall   wait_response           ; espera uma resposta
        jnb     b_timeout,conect143
conect141:
        lcall   cpowd1
        mov     v_loop,#30
conect142:
        lcall   tempo500ms
        djnz    v_loop,conect142
        ljmp    conecta_gprs            ; EDIT

conect143:
        movx    a,@dptr
```

```
        cjne    a,#'+',conect_140    ; se não for a resposta esperada, tenta de novo
        inc     dptr
        movx    a,@dptr
        cjne    a,#'C',conect_140    ; se não for a resposta esperada, tenta de novo
        inc     dptr
        movx    a,@dptr
        cjne    a,#'O',conect_140    ; se não for a resposta esperada, tenta de novo
        inc     dptr
        movx    a,@dptr
        cjne    a,#'P',conect_140    ; se não for a resposta esperada, tenta de novo
        inc     dptr
        movx    a,@dptr
        cjne    a,#'S',conect_140    ; se não for a resposta esperada, tenta de novo
        inc     dptr
        movx    a,@dptr
        cjne    a,#':',conect_140    ; se não for a resposta esperada, tenta de novo
        inc     dptr
        movx    a,@dptr
        cjne    a,#' ',conect_140    ; se não for a resposta esperada, aciona led
indicando erro
        inc     dptr
        movx    a,@dptr
        cjne    a,#'0',conect_140    ; se não for a resposta esperada, aciona led
indicando erro
        inc     dptr
        movx    a,@dptr
        cjne    a,#',',conect_140    ; se a resposta acaba aqui, verifica se tem
SIMCard     ; era conect_142
        inc     dptr
        movx    a,@dptr
```

```
        cjne    a,#30h,conect_140 ; se não for a resposta esperada, envia o comando novamente até encontrar algo útil
        inc     dptr
        movx    a,@dptr
        cjne    a,#',',conect_140
        inc     dptr
        movx    a,@dptr
        cjne    a,#"",conect_140
        inc     dptr
        movx    a,@dptr
        cjne    a,#'B',conect_150   ; Verifica se a primeira letra do nome da operadora é 'B', se não, tenta a próxima...
        inc     dptr
        movx    a,@dptr
        cjne    a,#'r',conect_140   ; Se sim, verifica se a segunda letra é 'r'. Se sim, considera que é Oi (Brasil Telecom)
        mov     v_operadora,#01
        jmp     conect_300

conect_150:
        cjne    a,#'T',conect_160   ; Verifica se a primeira letra do nome da operadora é 'T', se não, tenta a próxima...
        inc     dptr
        movx    a,@dptr
        cjne    a,#'I',conect_155   ; Se sim, verifica se a segunda letra é 'I'. Se sim, considera que é TIM BRASIL.
        mov     v_operadora,#02
        jmp     conect_300
conect_155:
        cjne    a,#'e',conect_181   ; Se não for 'I', verifica se é 'e'. Se sim, considera que é Telemig Celular.
```

```
        mov   v_operadora,#03
        jmp   conect_300
conect_160:
        cjne  a,#'C',conect_170  ; Verifica se a primeira letra do nome da operadora é 'C', se não, tenta a próxima...
        inc   dptr
        movx  a,@dptr
        cjne  a,#'l',conect_165  ; Se sim, verifica se a segunda letra é 'l'. Se sim, considera que é Claro.
        mov   v_operadora,#04
        jmp   conect_300
conect_165:
        cjne  a,#'T',conect_181  ; Se não for 'l', verifica se é 'T'. Se sim, considera que é CTBC Cellular.
        mov   v_operadora,#05
        jmp   conect_300
conect_170:
        cjne  a,#'A',conect_180  ; Verifica se a primeira letra do nome da operadora é 'A', se não, tenta a próxima...
        inc   dptr
        movx  a,@dptr
        cjne  a,#'m',conect_181  ; Se sim, verifica se a segunda letra é 'm'. Se sim, considera que é Amazonia Celular.
        mov   v_operadora,#06
        jmp   conect_300
conect_180:
        cjne  a,#'S',conect_181  ; Verifica se a primeira letra do nome da operadora é 'S', se não, tenta a próxima...
        inc   dptr
        movx  a,@dptr
```

```
        cjne    a,#'E',conect_181   ; Se sim, verifica se a segunda letra é 'E'. Se sim, considera que é SERCOMTEL.
        mov     v_operadora,#07
        jmp     conect_300
conect_181:
        ljmp    conect_140          ; Se não for nenhuma, chama COPS de novo pra conferir
conect_300:
        mov     v_timeout,#10
conect_310:
        lcall   limpa_buffer
        lcall   creg                ; Verifica a presença de rede
        lcall   wait_response
        jnb     b_timeout,conect3101
        setb    b_restart_modem
        ret
conect3101:
        mov     dptr,#z_buffer+11
        movx    a,@dptr
        cjne    a,#31h,conect_311   ; Se 1, registrado, rede local
        mov     v_timeout,#10
        jmp     conect_320
conect_311:
        cjne    a,#35h,conect_310   ; Se 5, registrado, em roaming
conect_320:
        lcall   limpa_buffer
        lcall   cpin
        lcall   wait_response
        jnb     b_timeout,conect321
        setb    b_restart_modem
        ret
```

```
conect321:
        mov     dptr,#z_buffer+9
        movx    a,@dptr
        cjne    a,#'R',conect_320
        inc     dptr
        movx    a,@dptr
        cjne    a,#'E',conect_320
        inc     dptr
        movx    a,@dptr
        cjne    a,#'A',conect_320
        mov     v_timeout,#10
        jmp     conect_330
con_timeout2:
        setb    b_timeout
        lcall   cpowd1
        mov     v_loop,#30
conect_3333:
        lcall   tempo500ms
        djnz    v_loop,conect_3333
        ljmp    conecta_gprs
conect_330:
        lcall   limpa_buffer
        lcall   csq
        lcall   wait_response
        jnb     b_timeout,conect331
        setb    b_restart_modem
        ret
conect331:
        lcall   tempo100ms
        mov     dptr,#z_buffer+9
        movx    a,@dptr
```

```
        cjne    a,#',',conect_340     ; se for ',', o valor está abaixo de 10, portanto
inaceitavel.
        jmp     conect_330            ; tenta de novo
conect_340:
        mov     v_timeout,#10
        lcall   limpa_buffer
        lcall   cstt
        lcall   wait_response
        jnb     b_timeout,conect341
        setb    b_restart_modem
        ret
conect341:
        movx    a,@dptr
        cjne    a,#'O',con_erro1      ; Verifica se foi OK
        lcall   tempo100ms
        inc     dptr
        movx    a,@dptr
        cjne    a,#'K',con_erro1      ; Verifica se foi OK
        lcall   tempo500ms
conect_350:
        mov     v_timeout,#10
        lcall   limpa_buffer
        lcall   cdnsorip
        lcall   wait_response
        jnb     b_timeout,conect351
        setb    b_restart_modem
        ret
conect351:
        movx    a,@dptr
        cjne    a,#'O',con_erro1      ; Verifica se foi OK
        lcall   tempo100ms
```

```
        inc     dptr
        movx    a,@dptr
        cjne    a,#'K',con_erro1      ; Verifica se foi OK
        lcall   tempo500ms
        mov     v_timeout,#15
conect_360:
        lcall   limpa_buffer
        lcall   ciicr
        lcall   wait_response
        jnb     b_timeout,conect361
        setb    b_restart_modem
        ret
conect361:
;       movx    a,@dptr
;       cjne    a,#'O',con_erro1      ; Verifica se foi OK
;       lcall   tempo100ms
;       inc     dptr
;       movx    a,@dptr
;       cjne    a,#'K',con_erro1      ; Verifica se foi OK
        lcall   tempo500ms
        mov     v_timeout,#15
conect_370:
        setb    b_conectando
        clr     b_restart_ip
        mov     v_reset,#0
        lcall   limpa_buffer
        lcall   cifsr
        lcall   wait_response
        jnb     b_timeout,conect3701
        setb    b_restart_modem
        ret
```

```
conect3701:
;       inc     dptr
;       movx    a,@dptr
;       cjne    a,#'.',conect_371    ; vê se um dos 4 primeiros dígitos é '.', pra conferir se veio um IP mesmo
        jmp     conect_380
conect_371:
        inc     dptr
        movx    a,@dptr
        cjne    a,#'.',conect_372
        jmp     conect_380
con_erro1:
        lcall   cpowd1
        mov     v_loop,#30
conect_5555:
        lcall   tempo500ms
        djnz    v_loop,conect_5555
        ljmp    conecta_gprs
conect_372:
        movx    a,@dptr
        cjne    a,#'.',con_erro1
        lcall   tempo100ms
conect_380:
        mov     P1,#00
        lcall   tempo500ms
        lcall   tempo500ms
        mov     v_timeout,#30
        lcall   limpa_buffer
        lcall   cipstart
        lcall   wait_response
        lcall   tempo500ms
```

```
        jnb     b_timeout,conect381
        setb    b_restart_modem
        ret
conect381:
        movx    a,@dptr
        cjne    a,#'O',conect_500    ; Verifica se foi OK
        lcall   tempo100ms
        inc     dptr
        movx    a,@dptr
        cjne    a,#'K',conect_500    ; Verifica se foi OK
        inc     dptr
        inc     dptr
        inc     dptr
        inc     dptr
        inc     dptr
conect_390:
        mov     a,v_timeout
        jnz     conect_391
        setb    b_timeout
        ljmp    conect_500
conect_391:
        movx    a,@dptr

        cjne    a,#'C',conect_390    ; Verifica se foi CONNECT
        lcall   tempo100ms
        inc     dptr
        movx    a,@dptr
        cjne    a,#'O',conect_500    ; Verifica se foi CONNECT ;  mandar  pra  um
erro que fecha o ip e abre de novo
        lcall   tempo100ms
        inc     dptr
```

```
        movx  a,@dptr
        cjne  a,#'N',conect_500   ; Verifica se foi CONNECT
        lcall tempo100ms
        inc   dptr
        movx  a,@dptr
        cjne  a,#'N',conect_500   ; Verifica se foi CONNECT
        lcall tempo100ms
        inc   dptr
        movx  a,@dptr
        cjne  a,#'E',conect_500   ; Verifica se foi CONNECT
        lcall tempo100ms
        inc   dptr
        movx  a,@dptr
        cjne  a,#'C',conect_500   ; Verifica se foi CONNECT
        lcall tempo100ms
        inc   dptr
        movx  a,@dptr
        cjne  a,#'T',conect_500   ; Verifica se foi CONNECT

        clr   b_restart_modem          ; EDIT limpa flags quando conecta
        clr   b_restart_ip
        clr   b_ser_ok

        lcall tempo500ms
        lcall tempo500ms

        jmp   conect_800
conect_500:
        mov   v_timeout,#20
        lcall limpa_buffer
        jmp   conect_530               ; EDIT
```

```
        lcall   wait_response
        jnb     b_timeout,conect_510
        setb    b_restart_modem
        ret
conect_510:
        mov     R0,#z_string
        mov     a,#'C'
        mov     @R0,a
        inc     R0
        mov     a,#'L'
        mov     @R0,a
        inc     R0
        mov     a,#'O'
        mov     @R0,a
        inc     R0
        mov     a,#'S'
        mov     @R0,a
        inc     R0
        mov     a,#'E'
        mov     @R0,a
        mov     a,#5
        mov     b, #20h
        lcall   procura_string
        jz      conect_520
        lcall   procura_erro
        jz      conect_530
conect_520:
        ljmp    conect_380
conect_530:
        lcall   cpowd1
        mov     v_loop,#30
```

```
conect_531:
        lcall   tempo500ms
        djnz    v_loop,conect_531
        ljmp    conecta_gprs
conect_800:
        setb    b_gprs_ok
conect_900:
        ret

conect_error:

;essa rotina tem que mandar ipclose e depois ipstart de volta!!!

con_err100:
con_err900:
        ret

; Espera a resposta do gprs pra depois tratar
wait_response:
wait_010:
        mov     dptr,#z_buffer
wait_100:
        setb    p_led1
        lcall   tempo500ms
        clr     p_led1
        lcall   tempo500ms
        mov     a,v_timeout
        jz      waited_too_much
        movx    a,@dptr
        cjne    a,#0Dh,wait_100
        inc     dptr
```

```
wait_200:
        movx  a,@dptr
        cjne  a,#0Ah,wait_200
        inc   dptr
        jmp   wait_900
waited_too_much:
        setb  b_timeout
wait_900:
        lcall tempo100ms
        clr   p_led1
        lcall tempo100ms
        ret

procura_erro:
        mov   R0,#z_string
        mov   a,#'E'
        movx  @R0,a
        inc   R0
        mov   a,#'R'
        movx  @R0,a
        inc   R0
        movx  @R0,a             ; manda 'R' de novo
        inc   R0
        mov   a,#'O'
        movx  @R0,a
        inc   R0
        mov   a,#'R'
        movx  @R0,a
        mov   a,#5
        mov   b,#20h
        lcall procura_string
```

```
        jnz     proc_err100
        setb    b_err_modulo
proc_err100:
proc_err900:
        ret

;--------------------------------------;
;                Comandos AT                ;
;--------------------------------------;

inicio_comando:                 ;inicia um comando at
        mov     a,#'a'
        lcall   tx_serial
        mov     a,#'t'
        lcall   tx_serial
        mov     a,#'+'
        lcall   tx_serial
        mov     a,#'c'
        lcall   tx_serial
        ret

fim_comando:                    ;finaliza um comando at
        mov     a,#0Dh
        lcall   tx_serial
        mov     a,#0Ah
        lcall   tx_serial
        ret

cfun1:
        lcall   inicio_comando
        mov     a,#'f'
```

```
        lcall   tx_serial
        mov     a,#'u'
        lcall   tx_serial
        mov     a,#'n'
        lcall   tx_serial
        mov     a,#'='
        lcall   tx_serial
        jnb     b_zera_cfun,cfun_100
        mov     a,#30h              ;30h somado pra enviar em ascii
        lcall   tx_serial
        jmp     cfun_900
cfun_100:
        mov     a,#31h              ;30h somado pra enviar em ascii
        lcall   tx_serial
cfun_900:
        lcall   fim_comando
        ret

creg:
        lcall   inicio_comando
        mov     a,#'r'
        lcall   tx_serial
        mov     a,#'e'
        lcall   tx_serial
        mov     a,#'g'
        lcall   tx_serial
        mov     a,#'?'
        lcall   tx_serial
        lcall   fim_comando
        ret
```

```
cpin:
        lcall   inicio_comando
        mov     a,#'p'
        lcall   tx_serial
        mov     a,#'i'
        lcall   tx_serial
        mov     a,#'n'
        lcall   tx_serial
        mov     a,#'?'
        lcall   tx_serial
        lcall   fim_comando
        ret

csq:
        lcall   inicio_comando
        mov     a,#'s'
        lcall   tx_serial
        mov     a,#'q'
        lcall   tx_serial
        lcall   fim_comando
        ret

cstt:
        lcall   inicio_comando
        mov     a,#'s'
        lcall   tx_serial
        mov     a,#'t'
        lcall   tx_serial
        lcall   tx_serial               ; manda 't' duas vezes
        mov     a,#'='
        lcall   tx_serial
```

```
        lcall   send_apn
        lcall   fim_comando
        ret

cdnsorip:
        lcall   inicio_comando
        mov     a,#'d'
        lcall   tx_serial
        mov     a,#'n'
        lcall   tx_serial
        mov     a,#'s'
        lcall   tx_serial
        mov     a,#'o'
        lcall   tx_serial
        mov     a,#'r'
        lcall   tx_serial
        mov     a,#'i'
        lcall   tx_serial
        mov     a,#'p'
        lcall   tx_serial
        mov     a,#'='
        lcall   tx_serial
        mov     a,#31h
        lcall   tx_serial
        lcall   fim_comando
        ret

ciicr:
        lcall   inicio_comando
        mov     a,#'i'
        lcall   tx_serial
```

```
        lcall   tx_serial       ; manda 'i' duas vezes
        mov     a,#'c'
        lcall   tx_serial
        mov     a,#'r'
        lcall   tx_serial
        lcall   fim_comando
        ret

cifsr:
        lcall   inicio_comando
        mov     a,#'i'
        lcall   tx_serial
        mov     a,#'f'
        lcall   tx_serial
        mov     a,#'s'
        lcall   tx_serial
        mov     a,#'r'
        lcall   tx_serial
        lcall   fim_comando
        ret

cipstart:
        lcall   inicio_comando
        mov     a,#'i'
        lcall   tx_serial
        mov     a,#'p'
        lcall   tx_serial
        mov     a,#'s'
        lcall   tx_serial
        mov     a,#'t'
        lcall   tx_serial
```

```
        mov   a,#'a'
        lcall tx_serial
        mov   a,#'r'
        lcall tx_serial
        mov   a,#'t'
        lcall tx_serial
        mov   a,#'='
        lcall tx_serial
        mov   a,#""
        lcall tx_serial
        mov   a,#'t'
        lcall tx_serial
        mov   a,#'c'
        lcall tx_serial
        mov   a,#'p'
        lcall tx_serial
        mov   a,#""
        lcall tx_serial
        mov   a,#','
        lcall tx_serial
        lcall send_ip
        lcall fim_comando
        ret

cipclose:
        lcall inicio_comando
        mov   a,#'i'
        lcall tx_serial
        mov   a,#'p'
        lcall tx_serial
        mov   a,#'c'
```

```
        lcall   tx_serial
        mov     a,#'l'
        lcall   tx_serial
        mov     a,#'o'
        lcall   tx_serial
        mov     a,#'s'
        lcall   tx_serial
        mov     a,#'e'
        lcall   tx_serial
        lcall   fim_comando
        ret

cipstatus:
        lcall   inicio_comando
        mov     a,#'i'
        lcall   tx_serial
        mov     a,#'p'
        lcall   tx_serial
        mov     a,#'s'
        lcall   tx_serial
        mov     a,#'t'
        lcall   tx_serial
        mov     a,#'a'
        lcall   tx_serial
        mov     a,#'t'
        lcall   tx_serial
        mov     a,#'u'
        lcall   tx_serial
        mov     a,#'s'
        lcall   tx_serial
        lcall   fim_comando
```

```
        ret

cpowd1:
        lcall   inicio_comando
        mov     a,#'p'
        lcall   tx_serial
        mov     a,#'o'
        lcall   tx_serial
        mov     a,#'w'
        lcall   tx_serial
        mov     a,#'d'
        lcall   tx_serial
        mov     a,#'='
        lcall   tx_serial
        mov     a,#31h
        lcall   tx_serial
        lcall   fim_comando
        ret

cops:
        lcall   inicio_comando
        mov     a,#'o'
        lcall   tx_serial
        mov     a,#'p'
        lcall   tx_serial
        mov     a,#'s'
        lcall   tx_serial
        mov     a,#'?'
        lcall   tx_serial
        lcall   fim_comando
        ret
```

```
c_insert_com:
        mov     a,#'+'
        lcall   tx_serial
        lcall   tx_serial
        lcall   tx_serial
        ret

c_comunica:
        call    c_insert_com
        mov     a,#'a'
        lcall   tx_serial
        mov     a,#'t'
        lcall   tx_serial
        mov     a,#'o'
        lcall   tx_serial
        lcall   fim_comando             ;manda 0D 0A no final.
        ret

cmgf:
        lcall   inicio_comando
        mov     a,#'m'
        lcall   tx_serial
        mov     a,#'g'
        lcall   tx_serial
        mov     a,#'f'
        lcall   tx_serial
        mov     a,#'='
        lcall   tx_serial
        mov     a,#'1'
        lcall   tx_serial
```

```
        lcall   fim_comando
        ret

cmgs:
        lcall   limpa_buffer
        lcall   tempo100ms
        lcall   inicio_comando
        mov     a,#'m'
        lcall   tx_serial
        mov     a,#'g'
        lcall   tx_serial
        mov     a,#'s'
        lcall   tx_serial
        mov     a,#'='
        lcall   tx_serial
        lcall   use_number
        lcall   fim_comando
        ret

ciurc:
        lcall   inicio_comando
        mov     a,#'i'
        lcall   tx_serial
        mov     a,#'u'
        lcall   tx_serial
        mov     a,#'r'
        lcall   tx_serial
        mov     a,#'c'
        lcall   tx_serial
        mov     a,#'='
        lcall   tx_serial
```

```
        mov    a,#'0'
        lcall  tx_serial
        lcall  fim_comando
        ret

dns_server  db     'at+cdnscfg="8.8.8.8","8.8.4.4"$'

cdnscfg:
        push        dph
        push        dpl
        mov         dptr, #dns_server
cdns300:
        mov         a,#00
        movc        a,@a+dptr
        cjne        a, #'$' , cdns310
        jmp         cdns350
cdns310:
        lcall       tx_serial
        inc         dptr
        jmp         cdns300
cdns350:
        pop         dpl
        pop         dph
        ret

use_number:
        push        dph
        push        dpl
        mov         dptr,#num_cel
use_num300:
        mov         a,#00
```

```
        movc    a,@a+dptr
        cjne    a, #'$' , use_num310
        jmp     use_num350
use_num310:
        lcall   tx_serial
        inc     dptr
        jmp     use_num300
use_num350:
        pop     dpl
        pop     dph
        ret

send_imei:
        push    acc
        push    dpl
        push    dph
        lcall   tempo500ms
        lcall   tempo100ms
        mov     a,#02Bh
        lcall   tx_serial
        lcall   tx_serial
        lcall   tx_serial
        lcall   tempo500ms
        lcall   limpa_buffer
        clr     b_timeout
        mov     v_timeout,#10
        mov     a,#'a'
        lcall   tx_serial
        mov     a,#'t'
        lcall   tx_serial
        mov     a,#'+'
```

```
        lcall   tx_serial
        mov     a,#'g'
        lcall   tx_serial
        mov     a,#'s'
        lcall   tx_serial
        mov     a,#'n'
        lcall   tx_serial
        lcall   fim_comando
        lcall   wait_response
        lcall   tempo500ms
        jb      b_timeout,send_imei
        mov     a,#'a'
        lcall   tx_serial
        mov     a,#'t'
        lcall   tx_serial
        mov     a,#'o'
        lcall   tx_serial
        lcall   fim_comando
        lcall   tempo100ms
        mov     dptr,#z_buffer
        inc     dptr
        inc     dptr
        mov     b,#15
send_im100:
        movx    a,@dptr
        lcall   tx_serial
        inc     dptr
        djnz    b,send_im100
send_im900:
        pop     dph
        pop     dpl
```

```
        pop     acc
        ret

send_apn:
        push    dph
        push    dpl
        clr     a
       lcall    limpa_buffer
        mov     a,v_operadora
        mov     dptr, #apn_oi
        dec     a
        jz      send_apn200
        setb    p_led2
        lcall   tempo500ms
        lcall   tempo500ms
        mov     dptr, #apn_tim
        dec     a
        jz      send_apn200
        mov     dptr, #apn_telemig
        dec     a
        jz      send_apn200
        mov     dptr, #apn_claro
        dec     a
        jz      send_apn200
        mov     dptr, #apn_ctbc
        dec     a
        jz      send_apn200
        mov     dptr, #apn_amazonia
        dec     a
        jz      send_apn200
        mov     dptr, #apn_sercomtel
```

```
        dec     a
        jz      send_apn200
        mov     dptr, #apn_vivo
send_apn200:
        mov     a,#00
        movc    a,@a+dptr
        cjne    a, #'$' , send_apn310
        jmp     send_apn350
send_apn310:
        lcall   tx_serial
        inc     dptr
        jmp     send_apn200
send_apn350:
        pop     dpl
        pop     dph
        ret

send_ip:
        push    dph
        push    dpl
        mov     dptr,#server_ip
send_ip300:
        mov     a,#00
        movc    a,@a+dptr
        cjne    a, #'$' , send_ip310
        jmp     send_ip350
send_ip310:
        lcall   tx_serial
        inc     dptr
        jmp     send_ip300
send_ip350:
```

```
        pop     dpl
        pop     dph
        ret

envia_sms:
        lcall   tempo500ms
        lcall   cmgf
        mov     v_timeout,#30h
envia_sms100:
        lcall   tempo500ms
        lcall   cmgs
        lcall   wait_response
        mov     dptr,#z_buffer+2
        movx    a,@dptr
        cjne    a,#3Eh,envia_sms100
        lcall   texto_mensagem
        lcall   wait_response

texto_mensagem:
        push    dph
        push    dpl
        mov     a,v_mensagem
txt_msg010:
        cjne    a,#01,txt_msg020
        mov     dptr,#msg_alarme
        jmp     txt_msg300
txt_msg020:
        cjne    a,#02,txt_msg030
        mov     dptr,#msg_desloca
        jmp     txt_msg300
txt_msg030:
```

```
txt_msg300:
        mov     a,#00
        movc    a,@a+dptr
        cjne    a, #'$' , txt_msg310
        jmp     txt_msg350
txt_msg310:
        lcall   tx_serial
        inc     dptr
        jmp     txt_msg300
txt_msg350:
        mov     a,#1Ah
        lcall   tx_serial
txt_msg900:
        pop             dpl
        pop             dph
        ret

;---------------------------------------------------------------;
;       LISTA DE DOMINIOS DAS OPERADORAS, PARA CONEXÃO
        ;
;---------------------------------------------------------------;

apn_oi                  db      '"gprs.oi.com.br","oiwap","oioioi"$'
apn_tim                 db      '"tim.br","tim","tim"$'
apn_telemig     db      '"gprs.telemigcelular.com.br","celular","celular"$'
apn_claro       db      '"claro.com.br","claro","claro"$'
apn_ctbc        db      '"ctbc.br","ctbc","1212"$'
apn_amazonia            db      '"gprs.amazoniacelular.com.br","celular","celular"$'
apn_sercomtel           db      '"sercomtel.com.br","sercomtel","sercomtel"$'
apn_vivo        db      '"3g.vivo.com.br","vivo","vivo"$'
```

```
;-------------------------------------------------------------;
;                        MENSAGENS                            ;
;-------------------------------------------------------------;

ip_att          db      ' IP e porta atuais: $'
pede_ip                 db      ' Informe o novo IP e a porta do servidor: $'
confirma_ip     db      ' IP e porta modificados para: $'
ip_ok           db      ' IP Ok, desligue o modem ou pressiona qualquer tecla para
repetir$'
msg_alarme      db      ' Atencao! O alarme do veiculo disparou.$'
msg_desloca     db      ' Atencao! O veiculo foi movido.$'



;-------------------------------------------------------------;
;                        Header pacote http                   ;
;-------------------------------------------------------------;

http_l1                 db      'POST /grr20071585/recebe_posicao.php HTTP/1.1$'
http_l2                 db      'Host: eng.eletrica.ufpr.br$'
http_l3                 db      'Connection: keep-alive$'
http_l4                 db      'Content-Length: 65$'
http_l5                 db      'Content-Type: application/x-www-form-urlencoded$'
http_l6a        db      '&imei=$'
http_l6b        db      '&lat=$'
http_l6c        db      '&long=$'
http_l6d        db      '&hem1=S&hem2=W$'

gps_pack        db
        '$GPGGA,235858.512,2527.4815,S,04914.2846,W,1,03,15.8,923.6,M,1.2,
M,,0000*5C..$'
```

```
trata_gps:
        push   dpl
        push   dph
        push   acc
        mov    R0,#gps_pack
        mov    dptr,#z_bufgps
trt_gps100:
        mov    a,#00
        movc   a,@R0+a
        jmp    txt_msg300
trt_gps200:
        mov    a,#00
        movc   a,@R0+a
        cjne   a, #'$' , trt_gps300
        jmp    txt_msg350
trt_gps300:
        movx   @dptr,a
        inc    dptr
        inc    R0
        jmp    trt_gps200
trt_gps350:
        pop    acc
        pop    dph
        pop    dpl

        ret

valida_gps:
        mov    dptr,#z_bufgps+42
        movx   a,@dptr
```

```
        jz      valida800
        lcall   send_http
        setb    b_gpsok
        jmp     valida900
valida800:
        clr     b_gpsok
valida900:
        ret


send_http:
        push        dph
        push        dpl
        lcall       tempo500ms
        clr         a
      lcall         limpa_buffer
        setb        p_led2
        mov         dptr, #http_l1
        call        send_http200
        lcall       fim_comando
        setb        p_led1
        mov         dptr, #http_l2
        call        send_http200
        lcall       fim_comando
        setb        p_led0
        mov         dptr, #http_l3
        call        send_http200
        lcall       fim_comando
        clr         p_led2
        mov         dptr, #http_l4
        call        send_http200
```

```
lcall       fim_comando
clr         p_led1
mov         dptr, #http_l5
call        send_http200
lcall       fim_comando
lcall       fim_comando
lcall       fim_comando
lcall       fim_comando
clr         p_led0

mov         dptr, #http_l6a
call        send_http200
lcall       send_imei
jmp         send_http400

mov         dptr, #http_l6b
call        send_http200
mov         dptr,#z_bufgps+18
call        send_http500

mov         dptr, #http_l6c
call        send_http200
mov         dptr,#z_bufgps+30
call        send_http500
movx        a,@dptr
lcall       tx_serial

mov         dptr, #http_l6d
call        send_http200
lcall       fim_comando
clr         p_led0
```

```
send_http200:
        mov         a,#00
        movc        a,@a+dptr
        cjne        a, #'$' , send_http310
        jmp         send_http350
send_http310:
        lcall       tx_serial
        inc         dptr
        jmp         send_http200
send_http350:
        ret
send_http400:
        lcall       tempo500ms
        clr         p_led0
        clr         p_led1
        clr         p_led2
        pop         dpl
        pop         dph
        ret

send_http500:
        push        b
        mov         b,#9
send_http510:
        movx        a,@dptr
        lcall       tx_serial
        inc         dptr
        djnz        b,send_http510
        ret
```

```
;-------------------------------------------------------------;
;                 DADOS PARA COMUNICAÇÃO                      ;
;-------------------------------------------------------------;

server_ip     db     "'eng.eletrica.ufpr.br","80"$'
num_cel       db     "'4199999999"$'

;-----------------------------------------------------------------------------
;      Inicializa Temporizadores e Interrupcoes
;-----------------------------------------------------------------------------

inic_temp:
        mov   TH0,#0                    ; P/ int 0
        mov   TL0,#0
        mov   TH1,#256-2                ;EDIT: #0FEh PARA 57600 BAUDS (x2)
        mov   TL1,TH1                   ; #099h
        mov   TH2, #0F1h
        mov   TL2, #099h                ;overflow a cada 10ms
        mov   PCON,#80h                 ;  Com  TH1=FAh(250)  e
SMOD(PCON.7)=0, Baud Rate=9600 bauds, conforme equação
                                        ;  Com  TH1=FEh(254)  e
SMOD(PCON.7)=1, Baud Rate=57600 bauds, conforme equação
        mov   TMOD,#23h                 ;T0 em modo 3. T1 em modo 2, agindo
como BRG para o porta serial
        mov   IP,#020h                  ;  com  EN6T  ativado,  a  baudrate
DOBRA, ou seja, 115200 BAUDS
        mov   IPH, #10h
        mov   TCON, #055h               ;dispara contagem automatica
        setb  IT1                       ;Muda TCON de 51h pra 55h e aciona
int do LCB na borda de descida de clock
```

```
        mov   SCON, #0FCh                       ; Controle de serial
        clr   SM2                       ; Setado p/ Controle de
Interrupcao
        clr   SM0                       ; Setado p/ Modo 3 e Zerado p/
Modo 1
        clr   TB8

        mov   T2MOD, #00h               ;não permite down counter no Timer2
        mov   T2CON,#04h
        mov   IE, #0B1h

        setb  ES
        setb  ET0
        setb  ET2
        setb  EA          ; Habilita Interrupcoes

        setb  EX0
        clr   EX1
        ret


;       Most typically, the timer is configured in the auto-reload mode
;       (mode 2).  In this case, the baud rate is given as:
;
;                    smod
;                    2       Oscillator frequency
;        Baud rate =  -------  x --------------------
;                    32        12 x (256 - TH1)


;       ***************************************************************
```

```
;       *                                                               *
;       *       AQUI COMEÇAM AS ROTINAS DA COMUNICAÇÃO SERIAL
        *
;       *                                                               *
;       *****************************************************************

;*******************************************************************************************
****************************************
;*      Transmite um dado pela serial
;*      Entradas:
;*              a = dado a transmitir
;*******************************************************************************************
****************************************

tx_serial:
        mov     SBUF,a
        jnb     TI,$            ; Espera transmitir o byte
        clr     TI
        ret

;*******************************************************************************************
****************************************
;*                                      Tratamento da interrupcao da serial
;*******************************************************************************************
****************************************

int_serial:
        push    psw
        push    acc
        push    dph
        push    dpl
```

```
        jb      RI,ise_RI                       ; Filtra se interrupção por
envio ou recepção
        jmp     ise_fim
ise_RI:
        clr     RI      ; Zera RI, permitir novas interrupcoes
        jb      b_conectado, ise_dado
        setb    b_conectado
        mov     v_serial+1, #high(z_buffer)
        mov     v_serial, #low(z_buffer)
ise_dado:                                       ; Filtra se na etapa da
recepção de dados
        mov     dph,v_serial+1
        mov     dpl,v_serial                    ; Aponta p/ end. base a
receber
        mov     a,SBUF
        movx    @dptr,a                         ;grava o byte na
memoria
        inc     dptr
        mov     v_serial,dpl
        mov     v_serial+1,dph                  ; Guarda novo end.
livre p/ receber
ise_fim:
        mov     v_minuto, #5
        setb    b_ser_ok
        jb      b_verificando,ise_900
        mov     v_reset,#00
ise_900:
        pop     dpl
        pop     dph
        pop     acc
        pop     psw
```

```
        reti

;*******************************************************************************
;       Limpa o z_buffer de comunicação
;       Entradas:
;               z_buffer
;       Saidas:
;               z_buffer zerado
;       Destroi:
;               nada
;*******************************************************************************
limpa_buffer:
        push    acc
        push    b
        push    dph
        push    dpl

        mov     dptr,#z_buffer
        mov     a,#00h
        mov     b,#0C7h
lim_buf_loop:
        movx    @dptr,a             ; Limpa apenas os primerios 200
        inc     dptr
        djnz    b,lim_buf_loop
;       mov     b,#0FFh
;lim_buf_loop2:
;       movx    @dptr,a             ; Limpa apenas os primerios 256 * 2 bytes
;       inc     dptr
;       djnz    b,lim_buf_loop2
        mov     v_serial,#00
        mov     v_serial+1,#00
```

```
        clr     b_conectado
        clr     b_timeout

        pop     dpl
        pop     dph
        pop     b
        pop     acc
        ret

;*************************************************************************
;       Rotina de recepção de RX1
;       Entradas:
;
;       Saidas:
;             acc
;       Destroi:
;             a
;*************************************************************************
int_rx1:
        push   acc
        push   psw
        push   v_reg0
        push   dph
        push   dpl
        push   IE
        clr    EA
        jb      p_rx1, int_rx1900
        mov    v_serial+1, #high(z_bufgps)
        mov    v_serial, #low(z_bufgps)
int_rx1100:
        lcall   rx1
```

```
        mov   dph, v_serial+1
        mov   dpl, v_serial
        movx  @dptr, a
        inc   dptr
        mov   v_serial+1, dph
        mov   v_serial, dpl
int_rx1110:
        jnb   p_rx1, int_rx1100
        djnz  v_ser_ok, int_rx1110
        mov   v_ser_ok, #00

int_rx1900:
        pop   IE
        pop   dpl
        pop   dph
        pop   v_reg0
        pop   psw
        pop   acc
        reti

;*******************************************************************************
;       Rotina de recepção por RX1
;       Entradas:
;             pino RX1
;       Saidas:
;             acc
;       Destroi:
;             a
;*******************************************************************************
rx1:
        push  b
```

```
        push   psw
        mov    b, #08
        mov    a, #00
        lcall  tempo100us
rx1_100:
        nop
        jnb    p_rx1,rx1_110
        setb   C
rx1_110:
        nop
        jb     p_rx1,rx1_120
        clr    C
rx1_120:
        nop
        nop
        rrc    a
        lcall  tempo100us
        djnz   b, rx1_100
        mov    v_ser_ok, #50                ;era zero (???)
rx1_130:
        jnb    p_rx1, rx1_130
rx1_900:
        pop    psw
        pop    b
        ret

;---------------------------------------------------------------------
End
```